HEWLETT-PACKARD

Integral Personal Computer

Comprehensive Guide

Integral Personal Computer

Comprehensive Guide

Edition 1 December 1984

Reorder Number
00095-90025

Notice

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

© Copyright 1984, Hewlett-Packard Company.

This document contains proprietary information that is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another language without the prior written consent of Hewlett-Packard Company. The information contained in this document is subject to change without notice.

Restricted Rights Legend. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in paragraph (b)(3)(B) of the Rights in Technical Data and Software clause in DAR 7-104.9(a).

© Copyright 1979, 1980, 1983, The Regents of the University of California.

This software and documentation is based in part on the Fourth Berkeley Software Distribution under license from The Regents of the University of California.

© Copyright 1980, 1984, AT&T Technologies. All Rights Reserved.

Portable Computer Division
1000 N.E. Circle Blvd.
Corvallis, OR 97330, U.S.A.

Printing History **Edition 1** December 1984 Mfg. No. 00095-90024

Contents

Chapter 1: General Information

1-1 Introduction
1-3 Using This Guide
1-3 What's in the Chapters?
1-4 Using Other Documentation
1-5 System Description
1-5 Windows and Multitasking
1-7 PAM
1-8 File System
1-11 File Structure
1-12 Using RAM and Electronic Disc

Chapter 2: Keyboard and Display Control

2-1 Overview
2-1 Menu and Mouse Use
2-3 System Menu
2-4 Mouse Use
2-5 General Keyboard Use
2-5 Typing Keys
2-8 Function Keys
2-8 Editing Keys
2-10 System Keys
2-11 General Window Control
2-13 Shuffling Windows
2-13 Moving and Stretching Windows
2-15 Reversing Window Color
2-16 Saving Windows
2-16 Pausing Windows
2-16 Hiding and Showing Windows
2-18 Removing Windows
2-19 Alpha and Graphics Window Control
2-20 Alpha Window Control
2-21 Graphics Window Control

Chapter 3: Using PAM

3-1 Introduction
3-1 The PAM Window
3-5 Getting Around in PAM
3-8 File Manipulation with PAM
3-8 Opening Folders
3-9 Starting Programs
3-10 Stopping Program Files
3-10 Creating Files
3-10 Rereading Folders
3-11 Closing Folders
3-11 Creating Folders
3-12 Deleting Files and Folders
3-14 Moving Files
3-16 Moving Files Between Discs
3-17 Renaming Files and Folders
3-18 Copying Files
3-20 Copying Files Between Discs
3-21 Copying and Moving Files to the Electronic Disc
3-22 Accessing Files with *view*
3-24 Printing Files
3-25 Sending Text
3-26 Echo
3-27 Redirecting Input and Output
3-31 Connecting Programs with Pipes
3-33 Specifying Sequential Processing
3-35 Specifying Non-Sequential Processing
3-36 Command Line Quoting
3-37 Specifying Operations with Files
3-38 Script Files
3-40 *Autost* Files
3-40 Environment Files
3-47 Selecting the Environment with *getenv*

Chapter 4: Command and Program Dictionary

4-1 Introduction
4-3 Parameters
4-6 Online Documentation
4-6 In This Chapter
4-7 Commands and Programs

Chapter 5: Installing and Using Peripherals

5-1 Introduction
5-1 Steps to Using Peripherals
5-4 Connecting Peripherals
5-4 HP-IB Peripherals
5-8 Serial Peripherals
5-8 Installing a Driver
5-10 Using Peripherals
5-11 The dev Folder
5-12 Peripheral Disc Drives
5-17 Peripheral Printers
5-20 Peripheral Plotters

Appendixes & Index

A-1 **Glossary**

B-1 **Keyboard Maps**

C-1 **Character Set**

Internal Printer Control

D-2 Printer Control Codes
D-6 Printer Controls
D-8 Superscripts and Subscripts
D-9 Print Pitches
D-11 Bold Print
D-11 Underlining
D-12 Mixing Modes
D-12 Line Spacing
D-13 Perforation Skip
D-14 Page Length
D-14 Text Length
D-15 Overstrike
D-16 Wrap-Around Mode
D-16 Printing Direction
D-17 Graphics
D-18 Graphics Escape Sequences
D-19 Display Functions Mode
D-20 Self Test
D-20 Resets
D-21 Printer Control Sequence Summary

E-1 **Messages**

Using the Vi Editor

F-1 Introduction
F-1 Getting Started Using Vi
F-6 Using the Cursor Commands
F-8 Adding, Deleting and Correcting Text
F-15 Writing Text to a Disc File
F-16 Where To Find More Information on Vi

I-1 **Index**

1 General Information

Introduction

By now you should have discovered that the HP Integral Personal Computer is very powerful.* With its 768 kilobytes of internal memory (expandable to more than 5.5 megabytes using I/O expanders), built-in thermal ink jet printer, 3 1/2-inch disc drive, and powerful 68000 microprocessor, it works the way you do—on several different tasks at the same time—and where you do—in a variety of work locations. Its operating system is based on HP-UX, Hewlett-Packard's implementation of the powerful UNIX† operating system.

Certain programs are built into the computer. The Personal Applications Manager, PAM, is a program you will become most familiar with. PAM helps you easily access files and perform other important tasks.

Additional software and hardware products are available to take full advantage of the computer's capabilities. Examples of available hardware include modems, plotters, flexible and hard disc drives, and HP-IB compatible peripherals. Additional software is available for tailoring your own applications from existing software or creating your own advanced applications.

* We assume that you have worked your way through the tutor disc or the user's guide. If you haven't, we strongly recommend that you take the time to do so.

† UNIX is a trademark of AT&T Bell Laboratories.

Your computer is supported with computer languages, utilities, and documentation. Software development packages for BASIC and C are available. Several types of utilities are available, including a collection of advanced utilities that are used on other HP-UX computers and specialized utilities that are designed specifically for the computer.

Six pre-recorded discs are included with the Integral PC. They are:

- **Tutor Disc.** The tutor disc shows you how to operate the Integral PC.
- **HP-UX Commands Disc.** This disc contains the commands used by the operating system.
- **Utilities Disc.** The Utilities disc contains several programs that help you control the computer.
- **Standard Applications Disc.** This disc has several programs including a word processor, a graphic design program, and an adventure game.
- **System Disc.** This contains several additional programs to help you control the computer.
- **Diagnostic Disc.** This disc is used to locate the source of problems that might occur with the computer.

Using This Guide

This guide will help you take advantage of the powerful operating system of your Integral Personal Computer. Before reading this guide, you should be familiar with the procedures described in the user's guide and feel comfortable with the operations described on the tutor disc.

While this guide covers many of the computer's capabilities that can be accessed without programming, there are many other capabilities that can be used when programming languages or plug-in interfaces are included in the system. Information about these capabilities is covered in separate manuals.

What's in the Chapters?

The chapters and appendixes of this guide contain detailed information about using your computer. Specifically, they contain the following information:

Chapter 1: General Information. This chapter gives general information about the computer. It describes basic system concepts necessary for understanding the operation of the computer.

Chapter 2: Keyboard and Display Control. This chapter gives detailed information on using the keyboard and optional mouse to control the computer's windowed display.

Chapter 3: Using PAM. This chapter describes the management of files by using PAM.

Chapter 4: Command and Program Dictionary. This chapter presents reference information about the standard utility programs provided with the computer.

Chapter 5: Installing and Using Peripherals. This chapter describes how to install and use peripheral printers, plotters, and disc drives with the computer. Interface drivers are also described.

Appendix A: Glossary. A detailed glossary of terms used in this guide.

Appendix B: Keyboard Maps. Correlates specific keys on the keyboard with control sequences and extended characters.

Appendix C: Character Sets. Contains a tabular listing of the standard character set for the The Integral.

Appendix D: Internal Printer Control. Contains reference material on the built-in ink jet printer.

Appendix E: PAM Error Messages. Provides a detailed listing of the error messages that may occur in PAM.

Appendix F: Using the Vi Editor. Provides a short tutorial on the *vi* editor.

Using Other Documentation

The manuals shipped with your computer enable you to learn how to use it for specific tasks, such as running applications, and help you use the computer for some advanced operations. A single document does not tell you eveything about the computer.

Documentation in addition to that shipped with your computer is available to assist you in learning about the capabilities of the computer and about the HP-UX operating system in general. See your dealer or H.P. sales and service Office for more information.

System Description

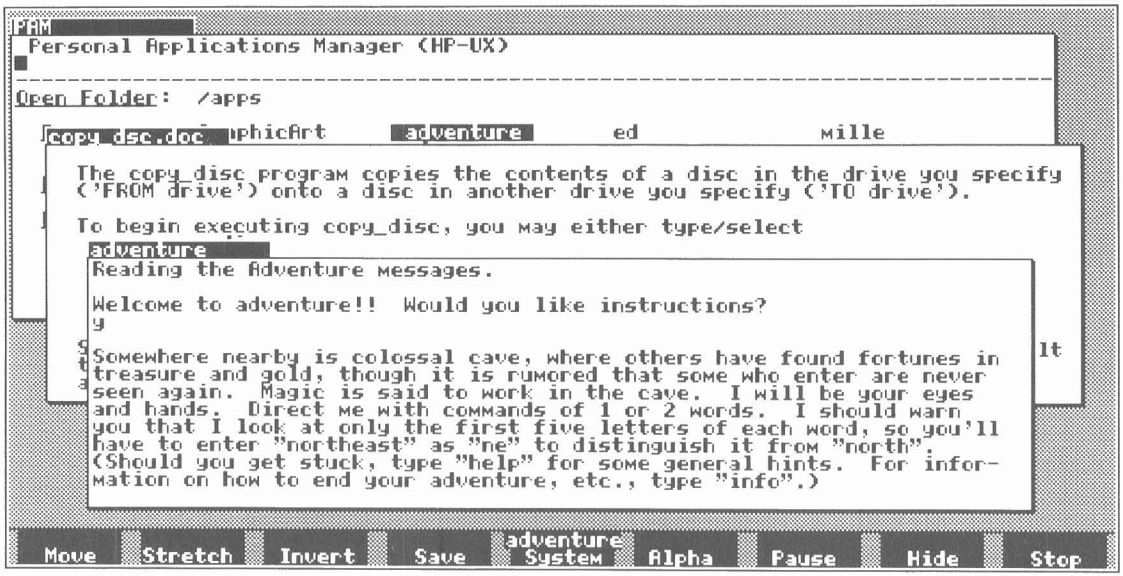
This section introduces some of the basics that govern the operation of the computer. Most of the material in this section is covered in greater detail in later chapters.

Windows and Multitasking

Perhaps the most striking difference between this computer and other computers you may have used is its multiple windows. Each window is an area on the computer display through which you communicate to a program (including PAM).

The computer's powerful operating system allows more than one program to run at a time. This is known as multitasking. When you run multiple applications, one or more windows may be displayed for each application. The following is a typical display showing multiple windows.

Display with Multiple Windows


PAM
Personal Applications Manager (HP-UX)
Open Folder: /apps
copy_disc.doc
adventure
ed
mille
The copy_disc program copies the contents of a disc in the drive you specify ('FROM drive') onto a disc in another drive you specify ('TO drive').
To begin executing copy_disc, you may either type/select
adventure
Reading the Adventure messages.
Welcome to adventure!! Would you like instructions?
y
Somewhere nearby is colossal cave, where others have found fortunes in treasure and gold, though it is rumored that some who enter are never seen again. Magic is said to work in the cave. I will be your eyes and hands. Direct me with commands of 1 or 2 words. I should warn you that I look at only the first five letters of each word, so you'll have to enter "northeast" as "ne" to distinguish it from "north". (Should you get stuck, type "help" for some general hints. For information on how to end your adventure, etc., type "info".)
Move
Stretch
Invert
Save
adventure System
Alpha
Pause
Hide
Stop

Each window maintains the information displayed by its associated program. Programs run independently and are visually separated on the display.

The keyboard and optional mouse are connected to only one window (or program) at a time. This window is known as the active window. The active window is the window identified in the middle of the menu.

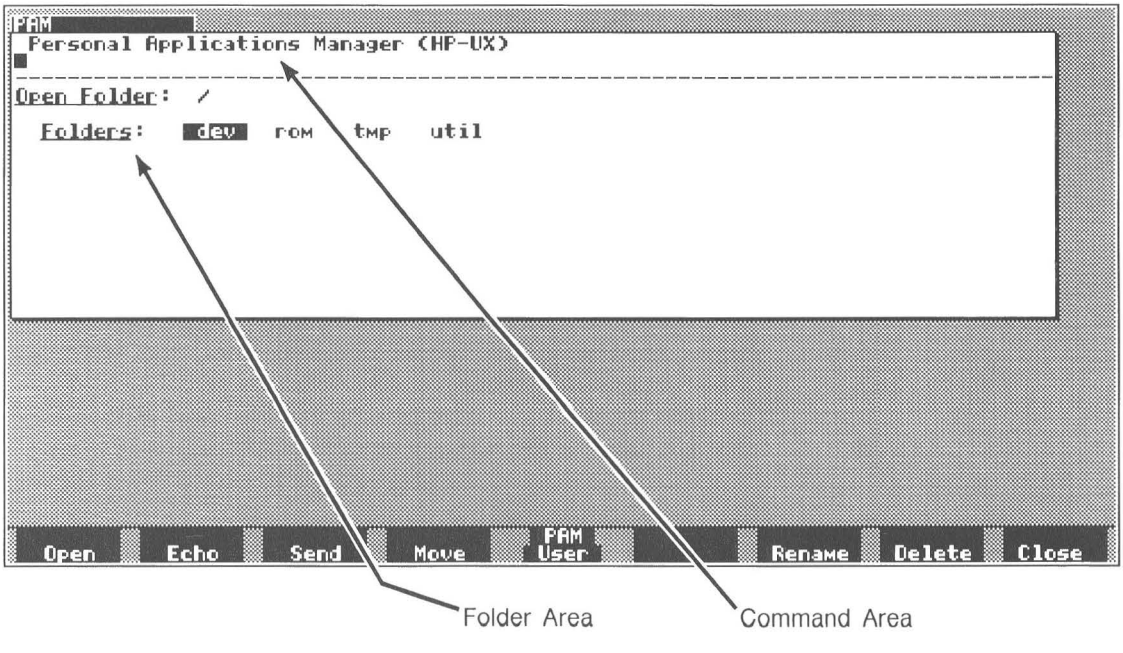
Windows can be manipulated in various ways, including being moved around the display, stretched, and hidden. Chapter 2 explains in detail how to control windows and their contents.

PAM

PAM, the Personal Applications Manager, is the part of the computer software you are probably the most familiar with by now. As its name says, PAM manages everything that has to do with running applications on the computer. As a matter of fact, PAM is an applications program itself.

Actually, PAM does a lot more than allow you to run applications programs. It is your interface to the power of the computer. So, it's worth the time to learn a little more about PAM. The screen below is a typical PAM display with some important features pointed out.

A Typical PAM Display



The PAM display is divided into two sections: the command area (the upper area) and the folder area (the lower area). Simply put, you enter commands in the command area and the folder area lists what's in a selected portion of the file system. Chapter 3 explains, in detail, how to use PAM to run applications, manage your files, and set up a new environment for the computer.

File System

As mentioned in the user's guide, the file system of your computer is similar to that of an office file cabinet. Each disc is like a drawer in the file cabinet, containing one or more folders which, in turn, can contain other folders or files. Let's examine this a little more closely.

Just What is a File? A file is a collection of information stored on a disc or in the computer's internal memory. Each file has a unique name. File names consist of from 1 to 14 characters. You can use any character of the alphabet or any numeral. You may also use the underbar, the minus, and in certain applications, the period. You should avoid using other characters because PAM attaches a special meaning to certain characters.

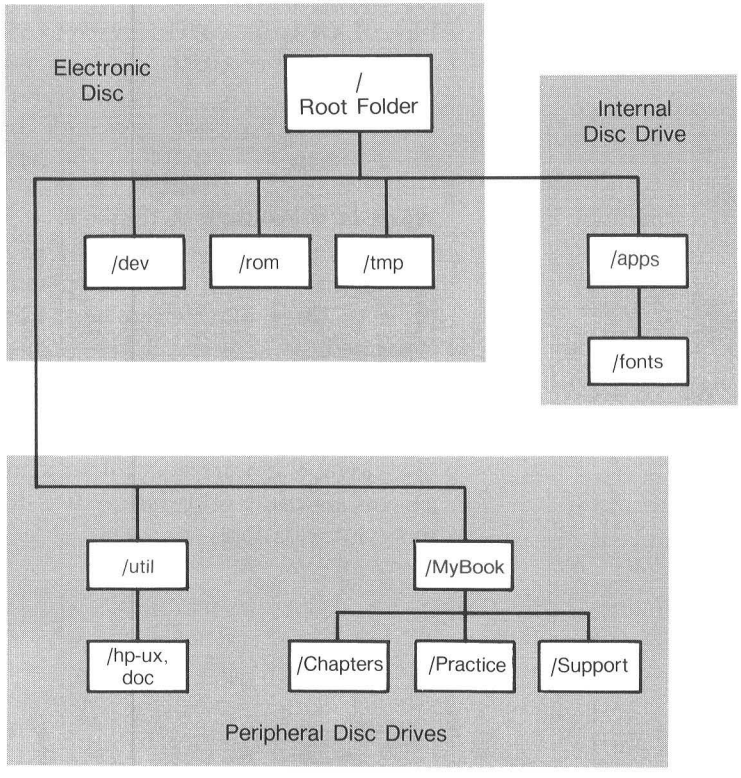
Typical file names include **Junelist**, **Memo-12**, and **Sep.Account**. PAM distinguishes between uppercase and lowercase letters. This means that **test**, **Test**, and **TEST** are three separate file names. If a file name is preceded by a period (**.**), the file is considered to be a hidden file. Hidden files are not listed in the folder area of the PAM display. However, a hidden file can still be accessed by typing in the complete file name (including the period).

There are four different types of files in the The Integral file system. Program files perform specific tasks. Typical types of program files include word processors, disc formatters, and spreadsheet programs. The data created by these files is stored as data files. These data files include spreadsheet and document files created by an application program. Another type of file is the folder, which is used to organize other files. The last type of file is a special file, the device file. Device files are used to access internal and external physical devices connected to the computer. Both folders and device files are discussed in more detail in the following sections.

What is a Folder? A folder is a collection of files. Folders help you organize your files by grouping them together. For example, if you are creating spreadsheets for three different projects, three folders can be made to keep the projects separated.

Each disc has one outermost file folder which holds all of the other files on the disc. When you insert a disc in the internal disc drive, the outermost folder is included in the file system. (If you are using an external disc drive, refer to chapter 5 for more information.)

The Integral PC file system is arranged as a hierarchy of folders. The top level of the file system is called the root folder. A typical file system might look like this (outermost folder names are in color):



File Structure

Accessing Files. The name you type to access a file is called the path name. The full path name gives the complete name of a file, starting at the root and tracing the hierarchy of the file. For example, to access the file **OpenMe**, you would refer to **/MyBook/PracticeFolder/OpenMe**. This tells the computer to look in the folder **MyBook**, then the folder **PracticeFolder** to find the file **OpenMe**; this is the path name for the file **OpenMe**. The first slash **/** character is the name of the root while the remaining slashes serve to separate the folder names in the path name.

You can also access a file by starting at a lower level in the hierarchy and using a relative path name. For example, if **MyBook** is the open folder, the relative path name to access **OpenMe** is **PracticeFolder/OpenMe**.

Displaying Folder Information. The open folder displayed in the PAM folder area provides certain information about the files it contains. Files are grouped according to type, such as program files and data files. Folders contain additional information about the files that are in them. (Refer to chapter 4 for detailed information.)

File Security. Every file has three permissions that tell the system what operations can be performed with the file: read, write, and execute. The specific operation each permission allows depends on the file type. Although these permissions generally default to allow you complete access, they can be changed with the *chmod* program. (Refer to chapter 4 for more information about changing file permissions.)

Device Files. Physical devices can be accessed through the use of device files. Each device file is associated with an internal or external device. Reading from or writing to a device file is equivalent to reading from or writing to a device. The **dev** folder in the root folder of the file system contains the device files.

When programs, commands, and utilities are run, certain devices/device files are accessed by default. These default device files used by the utilities are associated with internal devices. Peripheral disc drives are accessed with device files created automatically at power on. Device files and utilities for accessing other peripherals are also available from Hewlett-Packard.

Using RAM and Electronic Disc

The computer manages the use of its random access memory (RAM) dynamically. RAM is allocated to programs or files only when it is required. When a program has completed execution, the memory it used is returned to the system. This memory then is available for other programs and storing files in the root folder.

You can store files in RAM by using the electronic disc. The root folder is stored on the electronic disc. Therefore, to store files and folders in the electronic disc, you simply place them in the root folder. Since RAM is volatile, files stored in the root folder are lost when the computer is switched off. Using the root folder to store files or folders in RAM makes the following operations possible:

- High-speed program loading and file access from an application.
- Files can be copied between discs using the internal disc drive and the root folder for temporary storage while discs are exchanged.
- Disc-based utilities and commands can be copied to the root folder so the internal disc drive is available for accessing files stored on other discs.

The *copy* and *move* commands are used to place existing files in the root folder. You can also use the *makefolder* command to create a folder in the root folder. Instructions for using these commands are included in chapter 4.

It is possible to use up available RAM so that not enough memory is available for running programs and performing other needed operations. Storing files in the root folder and running several large programs at the same time can use up available RAM, for example.

The system displays an error message when the memory necessary to run a program or store a file in the root folder is not available. If there is not enough memory to run the program, you can choose to stop other programs so that sufficient RAM becomes available.

The following options allow you to increase the amount of RAM for running additional programs or storing files on the electronic disc. The options are ordered such that the first options have the least effect on other programs and files.

- Exiting from one or more application programs that are currently running.
- Waiting for a program to finish executing before starting another program. The RAM used by the first program is returned to system when it finishes executing and is available to run other programs.
- Removing a window you are finished using to make the memory used by the window and the program in it available to the system. Each alpha window normally uses a minimum of approximately 2K bytes with approximately 10K bytes being typical and graphics windows use approximately 16K bytes.
- Deleting files you have stored in the root folder. Do not remove any files or folders created by the system.
- Stopping programs with the *terminate* utility.
- Installing one or more memory modules to increase the amount of RAM available to the computer.

2 Keyboard and Display Control

Overview

This chapter expands on the display and window information presented in chapter 1. It describes how to use the menus, keyboard, and optional mouse to manipulate windows and the information displayed in them.

Menu and Mouse Use

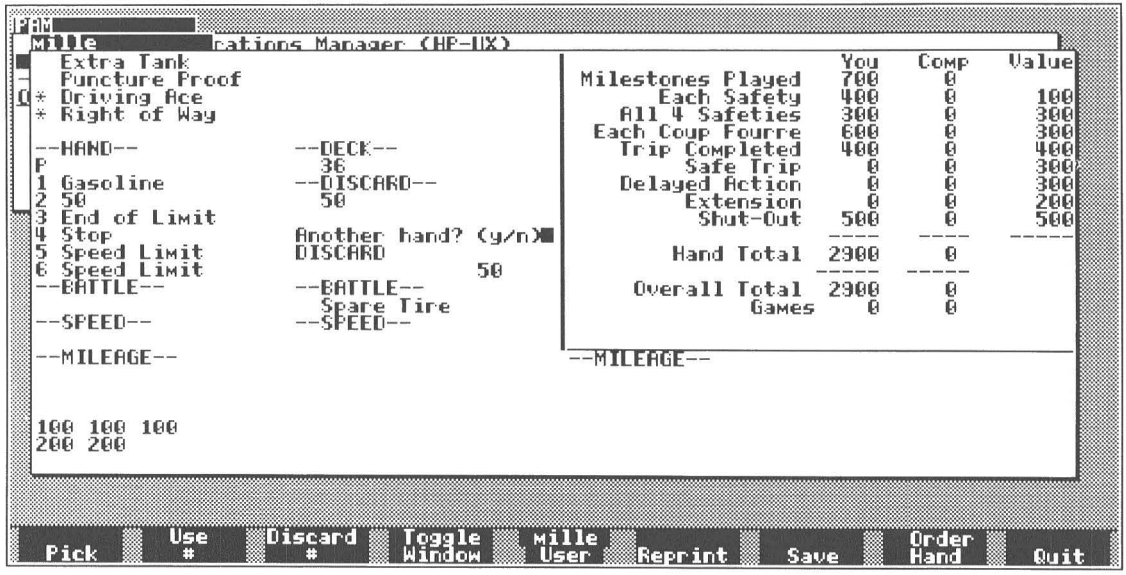
A menu is a horizontal inverse video area at the bottom of the display which contains function key descriptions. This area "floats" on top of any windows that might extend into the lower area of the screen. The menu does not modify the information in the window, even though the window is covered.

The (Menu) key on the keyboard and the right button on the mouse act as toggles, causing the menu to appear or disappear each time either key is pressed. Once a menu is displayed, it remains on the display until it is removed.

Whenever a window is active, a menu may show items appropriate for that window. This menu is called the *user menu.* There is also a system menu for commands that are used to control windows. The (User/System) key toggles between the user menu and the system menu. The center area of the menu identifies which menu is displayed. Menu items are selected with function keys (f1) through (f8) and shifted keys (f1) through (f8).

There is at most one menu on the screen, and it identifies the active window. Items in the menu are accessed by pressing the corresponding function key or by using the mouse (Select) key. And, even if the menu is not displayed (the (Menu) key was used to turn the menu off), the function keys can still be used to select the items that are assigned to them.

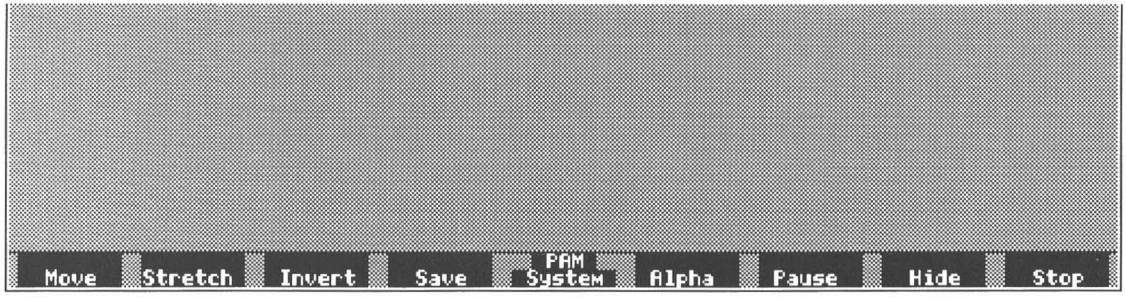
A Typical Menu



System Menu

The following system keys can be accessed by pressing (User/System) from any window. The changes made by these keys affect the active window.

System Menu



Mouse Use

The HP 46060A Mouse can be used to make choices in a menu without using the keyboard. The display pointer can be moved to any location on the screen by moving the mouse over a flat surface, such as a desktop. The left button on the mouse has the same function as the (Select) key on the keyboard. The right button has the same function as pressing (Menu). The display pointer can also be moved by using the (CTRL) key with the arrow keys.

Choices can be made by moving the display pointer to a menu label and pressing the left button on the mouse. In addition to menu selections, a window that is displayed but not active or a hidden window can be selected to be the active window with the display pointer. If the display pointer is over an area of the screen where it does not have a function, it is displayed as a diamond shape.

The computer must be turned off before you install the mouse. If you plug a mouse or keyboard into the computer while it is on, the computer may not recognize the new device.

The mouse connects to the computer through either one of the receptacles on the front of the computer. The computer can be used with both the mouse and keyboard plugged in, or it can be used with just a mouse or a keyboard. Instructions for installing and using a mouse with the computer are given in the user's guide.

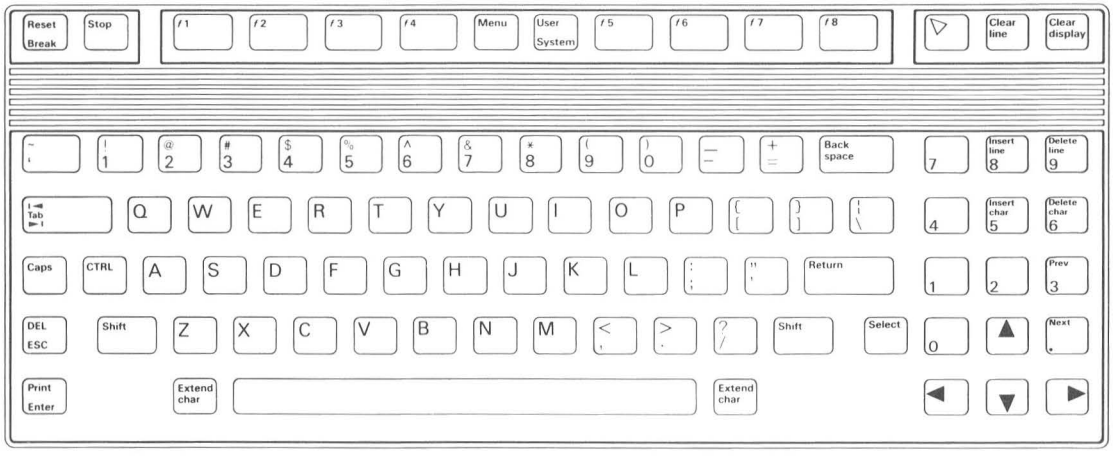
General Keyboard Use

The computer keyboard contains four groups of keys: typing keys, function keys, editing keys, and system keys. Some of the keys are particularly useful for window manipulation and are described in detail in the General Window Control section of this chapter.

Typing Keys

Some of the typing keys operate just like those on a standard typewriter keyboard, enabling you to type uppercase and lowercase letters, numbers, and symbols on the display. If you use a lowercase L for the number 1, the computer knows the difference. Also be careful with the number zero and the letter O.

Keyboard



You may not be familiar with the other typing keys, but they are useful for expanding system capabilities. The typing keys include:

Key	Operation
(Shift)	Use in combination with other keys to produce uppercase letters and to select any command or symbol on the upper half of a key. Pressing (Shift) while in uppercase (capitals) mode gives lowercase letters.
(Caps)	Makes all letters you type on the screen uppercase (capitals). Pressing (Caps) again returns to lowercase. Only the letter keys are affected.
(Tab)	Moves the cursor to the next tab stop.
(Shift) (Tab)	Moves the cursor to the previous tab stop.
(CTRL)	Used in combination with other keys to send control characters to the computer. For more information, refer to appendix B.
(Extend char)	Used to access the Roman Extension character set. For more information on these characters, refer to appendix C.
(Shift) (Caps)	Locks the (Extend char) key. Pressing a second time unlocks the mode.
(Return)	Indicates the end of a line of text or the end of a command.
(ESC)	Used in combination with other letters to send escape sequences to the computer. For more information, refer to "Using Control Sequences" later in this chapter.
(Shift) (DEL)	Generates the DEL character {■}.

Displaying Extended Characters. Using the (Extend char) key, you can display additional characters. These additional characters include symbols and foreign characters from the second half of the Roman8 character set table, diacritical marks, and symbols from the first half of the character set table that are displayed on foreign keyboards. Using (Shift)(Extend char), you can obtain upper-case foreign characters and other symbols. For example, to obtain the character ü, press (Extend char) and (=) simultaneously. To obtain the character Æ, press (Shift)(Extend char)(E) simultaneously.

To display a diacritical mark over a valid alpha character, you must use (Extend char) to obtain the mark and then type the alpha character. The table below shows the valid diacritical marks and corresponding keystrokes.

Valid Diacritical Marks and Keystokes

Diacritical Mark	Keystrokes
'	(Extend char)(R)
'	(Extend char)(T)
^	(Extend char)(Y)
··	(Extend char)(U)
~	(Extend char)(I)

Alpha characters that can be used with diacritical marks are: **a**, **e**, **i**, **o**, **u**, **y**, **A**, **E**, **I**, **O**, **U**, **Y**, **N**, and space, although not all combinations of these characters and diacritical marks are valid. For example, to type an **á** character, press (Extend char)(R) simultaneously and then press (a). If a non-valid character is typed after a diacritical mark, the mark is not displayed, but the character is.

Function Keys

The function keys (f1 through f8, unshifted and shifted) are used to access the functions shown across the bottom of the screen. (The display of functions across the bottom of the screen is the *menu.)* Keys f1 through f8 are used to access the bottom row of functions. The Shift key is used with keys f1 through f8 to access the underlined functions in the top half of the key label.

The function keys are defined by the application or program that is currently running. The Menu key turns the function (or system) key labels (referred to as the menu) off and on. The System/User key alternately displays the user and system function keys (those specifically for the current application and those specific to the system).

Editing Keys

The editing keys allow you to move the cursor around and add or delete text, thus enabling you to review what you have typed. Application programs (including PAM) may redfine some of these keys.

Key	Operation
◄, ►	Moves the cursor one space left or right.
▲, ▼	Moves the cursor one line up or down.
➤	Moves the cursor to the upper left corner of the active window (the *home position*).
Shift ➤	Moves the cursor to the bottom of text in the active window.
Shift ▲	Scrolls text up.
Shift ▼	Scrolls text down.
Clear line	Clears the current line from the cursor to the end of the line.
Shift Clear line	Clears the line containing the cursor.
Clear display	Clears all characters from the cursor to the end of text in the active window.
Shift Clear display	Clears all characters from the active window.
Back space	Moves the cursor back one space on the screen, erasing the character to the left.
Shift Insert line	Inserts a blank line at the position of the cursor; the following lines are moved down.
Shift Delete line	Deletes the line containing the cursor and moves all subsequent lines up.
Shift Insert char	Characters entered after this keystroke are inserted to the left of the cursor. Pressing these key again turns off the insert feature.
Shift Delete char	Deletes the character under the cursor and moves the remaining characters to the left.
Shift Previous	Used to see the previous page of current text.
Shift Next	Used to see the next page of current text.

System Keys

The system keys enable you to control computer operations manually. The system keys are:

Key	Operation
(Break)	Used in data communications to send a break to the host computer.
(Shift)(Reset)	Stops all programs and removes all windows, then displays PAM window.
(Stop)	Used to stop a program and remove its window.
(Shift)(Stop)	Used to abruptly stop a program and remove its window.
(Menu)	Turns the function or system key labels off and on. Even though the labels are not displayed, the current function or system keys are still active.
(System)	Toggles between the system and user function key labels (the menu).
(Shift)(User)	Selects the user funtion key labels.
(Select)	Used to select the choice indicated by the display pointer. Corresponds to the (Select) key on the mouse.
(Shift)(Select)	Used to shuffle windows on the display screen.
(Enter)	Sends the line that the cursor is on to the application. Also used in digitize mode of the graphics window.
(Shift)(Print)	Prints an exact copy of the screen image on the internal printer.
(CTRL)(Select)	Used to display a window if you can't display it another way.

Key	Operation
CTRL ◄ CTRL ►	Moves the display pointer left or right, respectively.
CTRL ▲ CTRL ▼	Moves the display pointer up or down, respectively.
CTRL ▼ ►	Moves the display pointer diagonally toward the lower right corner.
CTRL ▼ ◄	Moves the display pointer diagonally toward the lower left corner.
CTRL ▲ ◄	Moves the display pointer diagonally toward the upper left corner.
CTRL ▲ ►	Moves the display pointer diagonally toward the upper right corner.

General Window Control

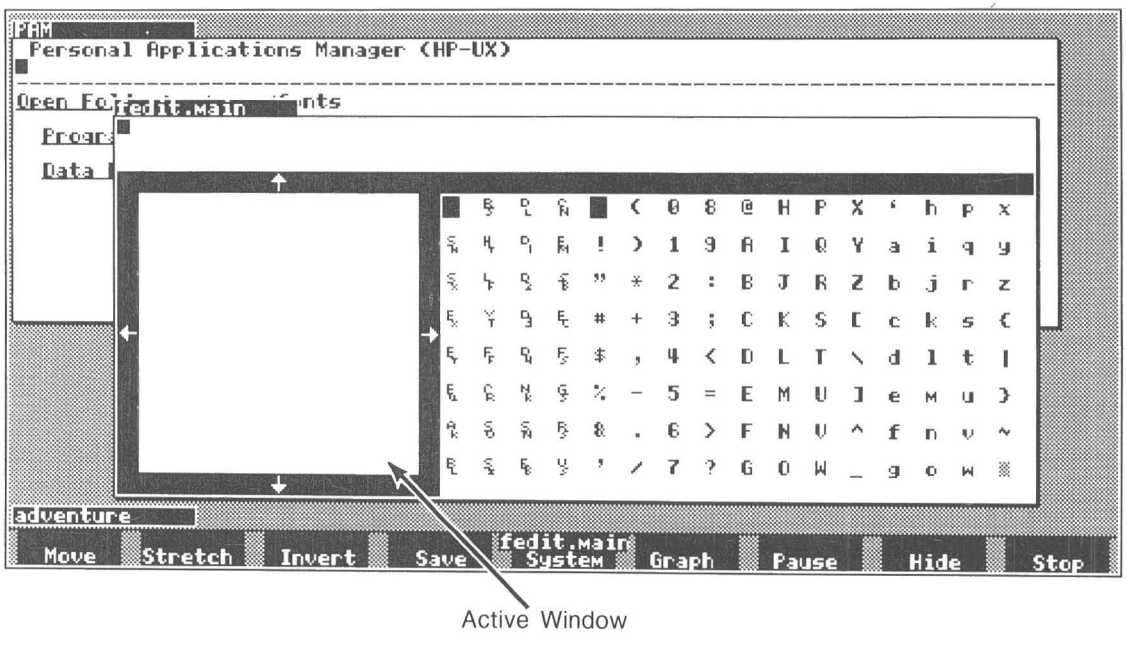
The keyboard is connected to only one window at a time. For example, when you type, the letters appear in one window. This window is called the active window. The center area of the menu at the bottom of the display identifies the active window.

You control the window that is active. A window is made active in the following ways:

- When you press Shift Select until the window you're interested in becomes active.
- When it is created by PAM or some other program.
- When the mouse is used to move the display pointer over any portion of the desired window and the Select button is pressed.

In most cases, the active window and the front-most window are the same. You can check the window name against the menu name as in the following screen.

The Active Window



Most window control is exercised by using the function keys of the System Menu. The following sections describe how to control the appearance of windows on the display.

Shuffling Windows

Since windows are allowed to cover each other, there must be a way of uncovering them. The easiest way to uncover a particular window is to use the (Shift)(Select) keystroke. This keystroke takes the back-most window, moves it to the front, and makes it the active window unless this is not allowed by the program. (For example, the *time_of_day* program does not allow its window to be displayed.) This is much like rearranging a pile of papers on your desk top. The (Shift)(Select) keystroke should be repeated until the desired window is in front.

Moving a window to the front does not change its contents. Only its visibility relative to the other windows is changed.

Moving and Stretching Windows

The location of the active window can be changed to any area of the screen by specifying a new location for the upper left corner of the window. This corner of the window is called the *anchor*. The size of the window remains the same during a move unless the anchor location causes part of the window to be off the screen.

Pressing **Move** ((f1)) changes the image of the display pointer to an angle bracket at the upper left corner of the window. This image is used to specify the anchor location. The display pointer can be moved with the mouse or with (CTRL) in combination with the arrow keys.

After the display pointer is at the new location, press (Select) or the left mouse button to move the window to the new location.

If you decide not to move a window after pressing **Move**, press any other key to cancel the move operation and leave the window at its current location. The key pressed to cancel the move operation performs its normal function. Pressing (Space) to cancel a move, for example, also sends a space to the window.

The size of the displayed portion of a window can be changed with a move operation. This occurs if the window anchor is moved to an area that would cause part of a window to be off the screen. The specified location is used for the upper left corner of the window, but the portion of the window that is off the screen is not displayed. The information in the window that is off the screen can be displayed by pressing (Shift) with the arrow keys to scroll the window contents.

Windows can be stretched or changed in size by selecting a new location for the lower right corner of the window. When **Stretch**, is pressed, the image of the display pointer changes to a bracket at the lower right of the window. The display pointer is moved with the mouse or with (CTRL) and the arrow keys. When a location is selected, the window size is adjusted to the new lower right corner location. The location of the upper left corner of the window does not change.

Depending on the location selected with the display pointer, the window size is increased or decreased.

Reversing Window Color

The background color of windows can be reversed so that light characters are displayed on a black background or black characters are displayed on a light background. If the reverse background is selected, an asterisk is displayed in the **Invert** (f3) menu label.

The background color selected in the PAM window is used as the default window color. If the backgound color of the PAM window is black, for example, then all programs and files accessed are displayed in windows with black backgrounds.

Saving Windows

Windows that contain information you want to retain can be saved. Windows that are not saved and are not being used by a program normally remain on the screen until another window is created. Use the Save key (f4) to select whether windows are preserved or are automatically removed when another window is created. An asterisk will appear in the Save (f4) menu label when the Save feature is activated.

Windows that are currently saved can be "unsaved" by pressing Save (f4) a second time when the saved window is the active window. This causes the asterisk in the menu label to disappear. To determine what windows are saved, shuffle the windows with the system menu displayed to see when the asterisk is present in the Save (f4) menu label.

Pausing Windows

The Pause key (f6) suspends output to the active window. If output is suspended to the active window, then an asterisk is displayed in the menu label. Pressing Pause (f6) again while the window is active causes output to resume.

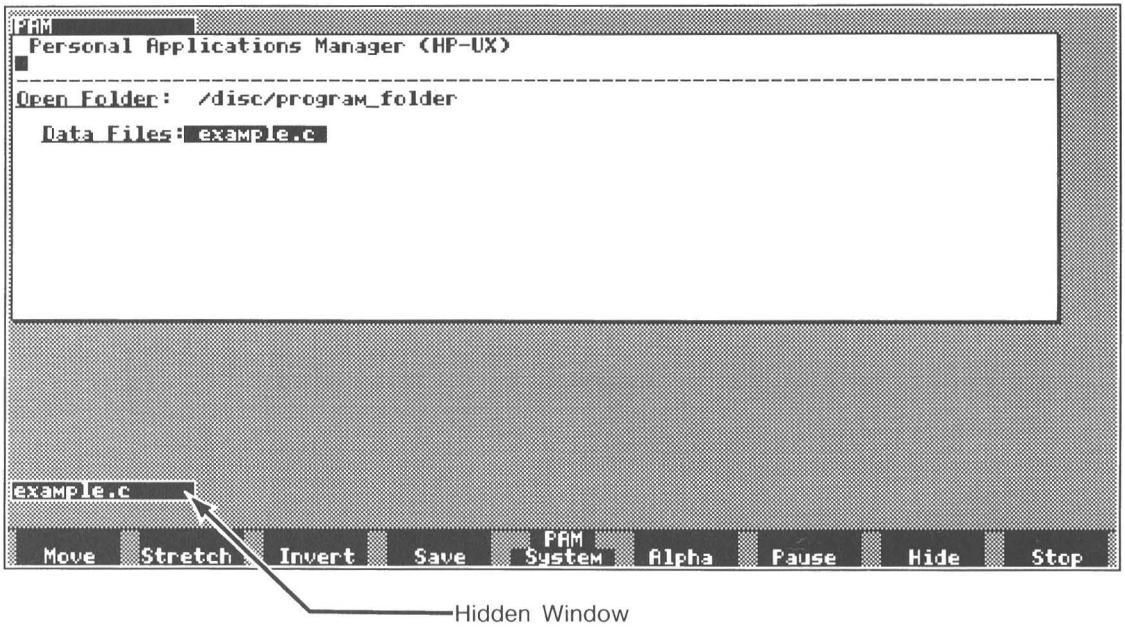
When a window is paused, the program producing the output continues to run until its buffer is filled. The program then suspends so that no additional output is produced until output is resumed to the window. When the program is resumed, no information is lost.

Hiding and Showing Windows

Windows can be hidden or removed from the screen if they are not currently needed. (The program can continue to run and information can continue to scroll on to the hidden window.) Hiding a window places it off-screen and does not change its contents. Windows can then be retrieved when they are needed. When a window is hidden, the window banner remains on the screen to remind you that the window still exists. The active window can be hidden by pressing Hide (f7).

The banner of each hidden window is displayed in the lower left corner of the screen. Any window that can be made the active window can be hidden.

Hiding a Window



Windows that have been hidden can be displayed again. The hidden window to be displayed is selected by moving the display pointer to the banner of the window and selecting it.

Removing Windows

When a window is removed, it is erased from the screen and the information in it is lost. Unused windows are automatically erased when other windows are created. You can remove windows manually to keep the "clutter" on the screen to a minimum and to reduce the number of times (Shift)(Select) has to be pressed when shuffling windows. You may also choose to remove windows when additional memory is needed for other programs.

Several options are available for removing windows manually besides the usual program exit procedure. If the program running in a window has an exit procedure, then this should be used before removing the window. If a program does not have a user exit procedure or is not responding to user commands, then you can choose to stop it with the keystrokes shown in the following table:

System Keys

Function	Description	Notes
(Stop) Stop	Signals the program using the active window to stop. The window is not removed until the program has stopped.	With application programs, this may have the same result as the user exit procedure.
(Shift)(Stop)	Forces the program using the active window to stop. The window is removed after the program stops.	Normal exiting procedures will not be allowed to complete.
(Shift)(Reset)	Stops all programs and removes all windows.	Any information in memory not stored in the RAM disc is lost.

Stopping a program may result in disc activity that is part of the normal exiting procedure for the program. Disc activity is indicated with the red light beside the disc drive. Additional steps to remove a window should be delayed until related disc activity stops.

Certain programs may not respond to (Stop). To stop these programs and remove the windows they are using, (Shift)(Stop) is used. Another option for stopping a specific program is to use the `terminate` utility (refer to chapter 4). This utility stops a program but leaves its window on the screen. After the program has stopped executing, (Stop) or Stop ((f8)) can be used to remove the window.

All windows are removed when the computer is reset. Resetting the computer with (Shift)(Reset) stops all programs.

Alpha and Graphics Window Control

Applications run on the Integral PC in either of two types of windows: an alpha window or a graphics window. When these windows are active, a third menu is available for use (the User and System are the other two menus). The third menu is accessed by pressing (f5) on the System Menu.

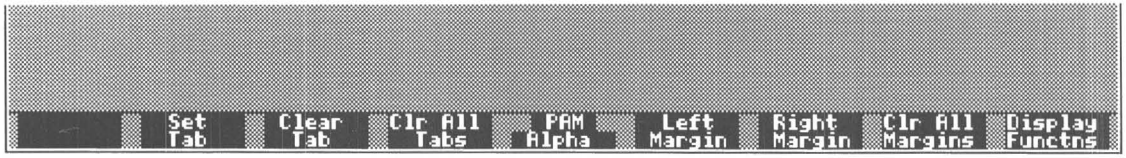
This section describes only those alpha and graphics window features that can be accessed from the computer keyboard.

Alpha Window Control

Alpha Window Menu. The function keys for controlling an alpha window are similiar to those used to control a computer terminal. Alpha window function keys are shown and described below.

These keys provide manual window configuration control which may be in conflict with the controls designed into a particular application. They are rarely needed in normal operation and are provided as an extra convenience for experienced terminal users.

Alpha Window Menu



- The **Set Tab** key (f2) sets a tab stop in the current cursor column. (Default tab stops are set every eight columns starting at column 0.)
- The **Clear Tab** key (f3) clears a tab stop from the current cursor column.
- The **Clr All Tabs** key (f4) clears all tab stops for the active window.
- The **Left Margin** key (f5) causes the left margin to be set at the current cursor column. (The default left margin is at column 0.)
- The **Right Margin** key (f6) sets the right margin at the current cursor column. (The default right margin is at column 79.)

- The Clr All Margins key (f7) clears the left and right margins to the default margin columns.
- The Display Functions key (f8) enables or disables display functions mode. Control codes and escape sequences are displayed, rather than being executed, in this mode. An asterisk in the menu label indicates display functions mode is activated.

Using Control Sequences. Control codes and escape sequences can be entered directly in an alpha window from the keyboard and have the same effect as if they originated from a program. Control codes are accessed by pressing CTRL with a character key. Escape sequences are entered by pressing ESC followed by several other characters. Unless the window is in display functions mode, the control codes and escape sequences are not displayed. (If you need more information about escape sequences, contact your local Hewlett-Packard dealer.)

Graphics Window Control

A menu is provided with graphic windows to perform some of the same functions as the control panel on a mechanical plotter. The function keys can be used to move the plotter pen up and down and change its speed, for example. You can draw limited graphics in an active plotter window with the arrow keys and the menu.

The maximum size of a graphics window is the size of the screen. The default window size is slightly smaller than the screen size. The screen is 18 by 9 centimeters and displays 512 by 255 dots. The dots are displayed with a uniform resolution of 2.8 dots per millimeter in both vertical and horizontal directions. The graphics window can be increased in size to display all the dots with the Stretch (f2) and Move (f1) function keys in the PAM system menu.

Graphics Window Menu. When a graphics window is the active window, you can display the graphics menu by pressing Graphics (f5) on the System Menu. The following menu is then displayed:

The Graphics Menu

- The Pen Up key (f1) puts the pen in the up position and changes the pen image to the outline of a pen. The pen up position allows you to move the pen without drawing a line.
- The Pen Down key (f2) puts the pen in the pen down position and changes the pen image to a hatched-in pen. Moving the pen in the pen down position causes a line to be drawn.
- The Show Pen key (f3) turns the pen image on or off. The tip of the pen corresponds to the plotter pen position. When the pen image is on, an asterisk is displayed in the Show Pen menu label.
- The Fast Pen key (f4) changes the relationship between the arrow keys and the plotter pen. In normal mode, the pen moves one dot each time you press an arrow key. If fast pen mode is selected, the pen moves eight dots each time an arrow key is pressed. Fast pen mode is indicated by an asterisk in the Fast Pen menu label.
- The ERROR key (f7) clears error conditions in the active graphics window. The menu label is only displayed when an error condition exists.
- The Clear key (f8) clears the graphics window. All data in the graphics window is lost.

Plotting From the Keyboard. The patterns that can be plotted in a graphics window directly with the keyboard are limited to lines in the horizontal, vertical, and diagonal directions. Plotting is accomplished by putting the plotter pen in the down position and then moving it across the window with the arrow keys.

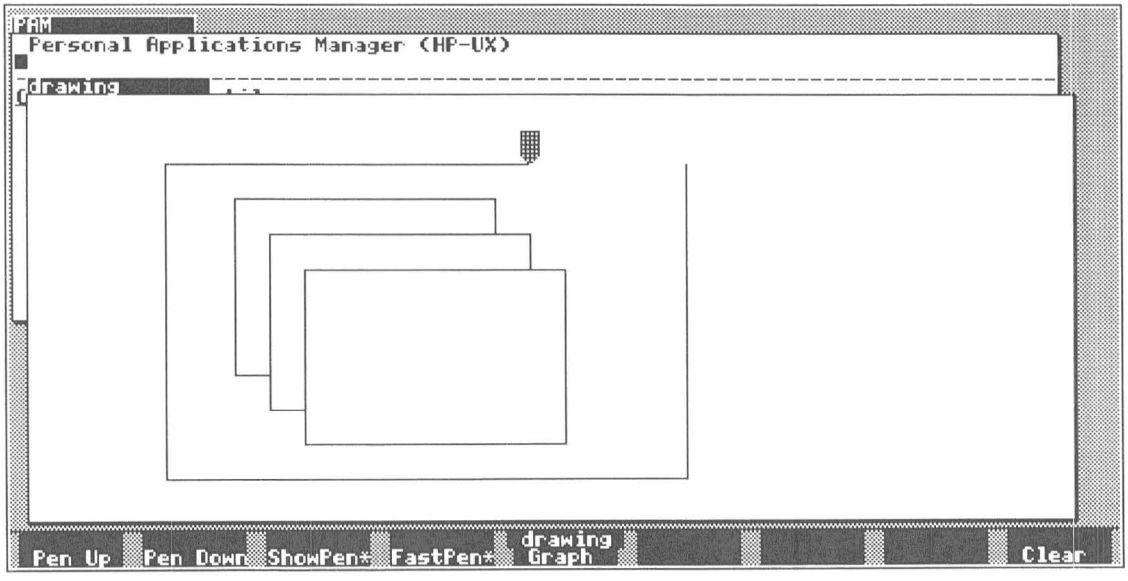
After creating a graphics window with the *plotter_is* utility (see chapter 4), the plotter pen is located in the lower left corner of the window. To display the pen, press **Show Pen** (f3) . The pen can be moved into view by pressing ➤.

Before you can draw a line with the plotter pen, it must be put in the pen down position. To do this, press **Pen Down** (f2) . The arrow keys can now be used to draw lines in the graphics window. The pen is moved like the cursor. Vertical lines are drawn with ▲ and ▼. Horizontal lines are drawn with ◄ and ➤. To draw a diagonal line, press a vertical and a horizontal arrow key.

Pressing an arrow key while a graphics window is active moves the pen one screen dot in normal mode. In fast pen mode, the pen moves eight dots when an arrow key is pressed. To change the pen speed, press **Fast Pen** (f4). If the plotter window is in fast pen mode, an asterisk is displayed in the menu label. Holding an arrow key down for several seconds causes buffering and may result in the pen continuing to move for a short distance after releasing the arrow key. Pressing Shift with the arrow keys rolls the contents of the active window.

Lines drawn in the graphics window cannot be erased selectively. When you want to begin a new drawing, the entire window can be cleared by pressing Clear (f8). If you are finished using the plotter window, it can be removed by selecting it as the active window and pressing Stop, Stop (f8) on the system menu), or Shift Stop.

Drawing a Line in the Graphics Window



3 Using PAM

Introduction

PAM, the Personal Applications Manager, is a built-in program that enables you to easily interact with the computer. It gives you control of various file system operations, such as renaming and copying files and starting applications programs.

The PAM Window

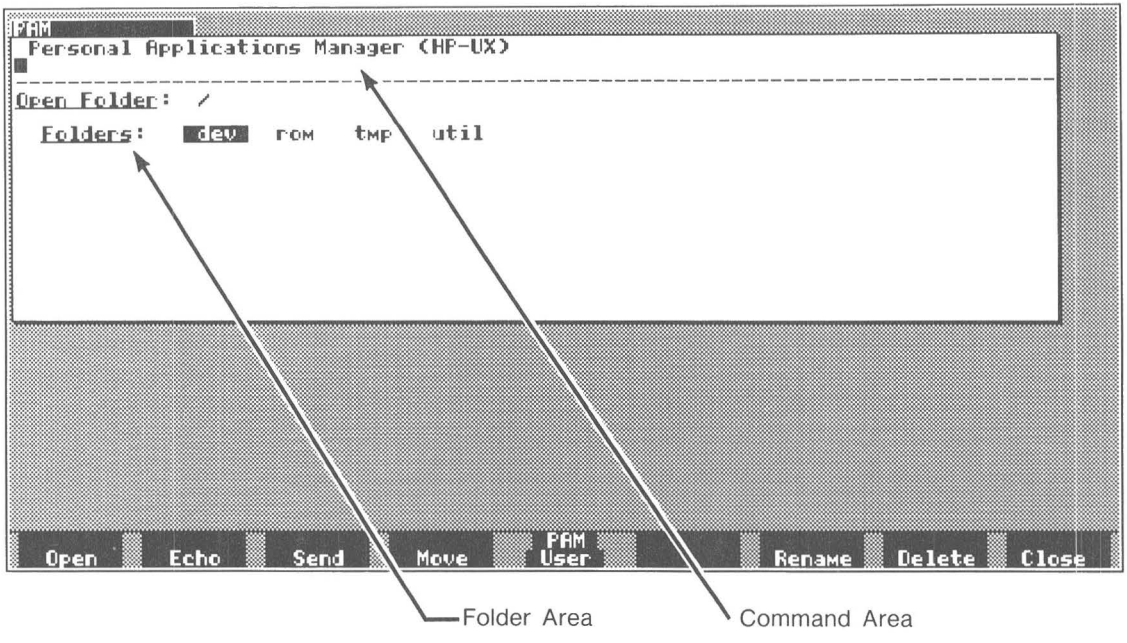
PAM is always available for use. Whenever the computer is turned on, a window is created in which PAM is running. Accessing PAM is simply a matter of selecting the PAM window (moving the PAM window in front of any other windows on the display).*

The PAM window is divided into two sections. The upper portion, the command area, is where you issue commands to and receive feedback from PAM. The lower portion, the folder area, is where PAM displays the names of files in the open folder.

* If there is a file named Autost in the outermost folder of the disc in the internal disc drive, the file is executed when the computer is first turned on, as though it was typed as the first command. The PAM window will not appear until the Autost program has completed.

A typical PAM window looks like this:

The PAM Window



The Command Area. The PAM command area is the portion of the screen where you issue commands and receive messages. The command area contains two lines. The first line is the message line and the second line is the command line.

PAM uses the message line to display informational messages. These are clear, simple messages indicating that a problem exists. In some cases, the message contains suggestions about how to fix the problem. The types of errors reported include syntax errors in commands and attempts to access files that do not exist.

Informational messages may also prompt you to complete a certain task. For example, PAM uses prompting when you copy a file. Through the message line, PAM tells you how to specify the destination of the Copy command. Appendix F contains a complete list of PAM messages and their meanings.

The command line allows you to enter commands directly from the keyboard. This lets you specify operations that involve files in folders other than the open folder and to enter commands that require parameters.

The Folder Area. The lower part of the PAM window, the folder area, tells you what is on the disc. The folder area shows your open folder and the other folders, program files, and data files in the open folder.*

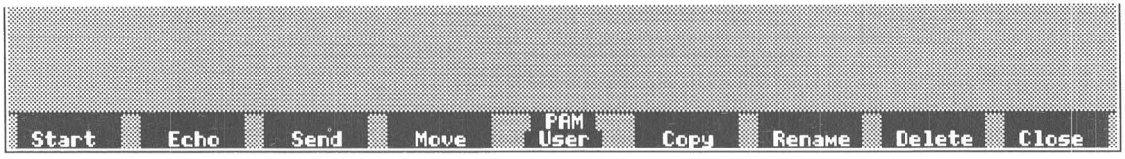
One file is highlighted in inverse video in the folder area. The highlighted file is the one on which you can perform operations using the User keys. Use the arrow keys to highlight a different file. (Tab) and (Shift)(Tab) can also be used to highlight a different file.

To keep the contents of the folder area current, the PAM window is updated each time you change your open folder, execute a PAM command that changes the information on the disc, or insert a new disc. However, it is possible for a program to change the disc without PAM knowing about it. Whenever you wish to know exactly what is on the disc, highlight the open folder and press Reread ((f1)) to update the folder area.

* The folder area shows only the first 99 files (or folders) stored in the open folder. If you have more than 99 files in the folder, use *ls* (described in chapter 4) to view the contents of the folder.

The User Menu. The PAM User Menu allows you to perform frequently-used file operations with a single keystroke. To execute a command, just press the specific function key. The PAM User Menu is summarized below.

The PAM User Menu



PAM User Menu Summary

Function Key	Menu Label	Description
f1*	Open	Opens the highlighted folder.
	View	Displays the highlighted data file.
	Start	Executes the highlighted program.
	Reread	Updates the open folder information. (The folder name must be highlighted.)
f2	Echo	Copies the highlighted file name to the command line.
f3	Send	Sends the highlighted file name to the next window down.
f4	Move	Used to move the highlighted file.
f5	Copy	Used to copy the highlighted file.
f6	Rename	Used to rename the highlighted file.
f7	Delete	Used to delete the highlighted file.
f8	Close	Closes the open folder.

* The menu label and function of f1 changes with the type of file being accessed.

Getting Around in PAM

Special PAM Keys. Certain keys perform special functions when the PAM window is active. For example, the (Tab) key moves the file indicator across a row of file names. The special PAM keys are summarized in the following table, all other keys retain their functions described in chapter 2.

Special PAM Key Summary

Key	Description
(▲)	Moves the file indicator up a column of file names.
(▼)	Moves the file indicator down a column of file names.
(►)	Moves the file indicator across a row of file names.
(◄)	Moves the file indicator back along a row of file names.
(↖)	Moves the file indicator to the first file in the open folder.
(Shift)(↖)	Moves the file indicator to the last file in the open folder.
(Tab)	Moves the file indicator across a row of file names.
(Shift)(Tab)	Moves the file indicator back along a row of file names.
(Shift)(Next)	Displays the next screenful of file names.
(Shift)(Prev)	Displays the previous screenful of file names.
(Return)	Executes the command in the command line. If the command line is empty, then pressing (Return) is the same as pressing (f1).
(Ctrl)(l)	Rereads the discs and updates the open folder information.
(Ctrl)(c)	Terminates and then restarts PAM. (This is the same as (Shift)(Stop).)

Using the PAM Command Line. As noted earlier, entering commands into the command line allows you to perform operations not on the PAM user menu. This also enables you to specify operations that involve files not displayed in the open folder.

To execute commands from the command line, make the PAM window active, enter the command, and press (Return). When (Return) is pressed, the command in the command line is executed. (If the command line is empty, then pressing (Return) is the same as pressing (f1).) If you enter a command containing an error, an error message is displayed. The keystrokes in the following table can be used to correct errors in the command line. After correcting the error, press (Return).

Command Line Editing Keys

Keystroke	Function
(Shift)(◄)	Moves the cursor one character position to the left.
(Shift)(►)	Moves the cursor to the right one character position.
(Shift)(▲)	Displays the previous command in command memory.
(Shift)(▼)	Displays the next command in command memory.
(Back space)	Moves the cursor one space to the left and erases the current cursor character.
(Clear line)	Clears all characters from the current cursor character to the right end of the line.
(Clear display)	Clears the command line and all lines following in command memory.
(Shift)(Clear line)	Clears all characters on the command line.
(Shift)(Delete line)	Clears the command line.
(Shift)(Delete char)	Deletes the current cursor character.
(Shift)(Insert char)	Turns insert character mode on or off.

The last 20 lines entered in the command line are kept in a portion of memory called *command memory*. These lines can be displayed by pressing (Shift) with the vertical arrow keys to the command line through command memory. For example, pressing (Shift)(▲) displays the previous command in command memory. This allows you to execute previously entered commands by moving the command line, making any necessary modifications, and pressing (Return) again.

When using the command line to specify operations, a single period (.) can be used to represent the path name of the open folder. Two adjacent periods (. .) specify the folder in which the open folder resides.

The command line can be used for executing commands that use parameters. Some commands require parameters, while other commands have optional parameters. The syntax shown for each command represents the different ways you can execute it. If one of the syntax shows no parameters, then parameters for the command are optional. Note that parameters must be separated with one or more spaces.

Many commands allow you to repeat either the first or second parameter several times to perform multiple operations. The *copy* command allows you to specify multiple files for copying to a folder by repeating the first parameter, for example. The ellipsis (. . .) indicates that the parameter it follows can be repeated several times. With the *copy* command, this allows you to enter the first parameter, *file_name*, several times to identify the files you want to copy to the folder you specify with the second parameter. The command form for the *copy* command that allows the first parameter to be repeated is shown as:

copy *file_name ... folder_name*

You substitute the actual names of files and folders for the parameters. By repeating the first parameter, the following command specifies the copying of three files, **productA**, **productB**, and **productC**, to the folder **/usr/execfiles**:

```
copy prodA prodB prodC /usr/execfiles
```

File Manipulation with PAM

To access folders and files, you must use PAM. To access a file in a folder open the folder and then select the file.

Opening Folders

Pressing the Open (f1) function key opens the highlighted folder. (This function key appears when the highlighted file is a folder.) If the highlighted folder cannot be opened, a message is displayed. Using the Open function key to open a folder is equivalent to entering the name of the highlighted folder in the command line and pressing Return. You can also move the display pointer with the mouse and press the select key on the mouse to open the folder.

The root folder of a disc is opened when that disc is inserted into the computer.

Using the *cd* command (change directory) you can change the current open folder. The folder to be opened can be specified with a simple name or a path name. If the folder is in the open folder, then a simple name is used. You can enter the command with or without a parameter.

```
cd
cd folder_name
folder_name
```

If the folder name is omitted, then the system attempts to open a folder called the *home folder*. If the home folder is not defined, then the root folder is opened. The home folder is part of the active environment and is defined by the home variable in the **/rom/.environ** file. The active environment is normally set at power on if the disc in the internal drive contains a hidden file called **.environ** in its uppermost folder or in **rom**. Additional information about setting the home folder is presented later in this chapter.

Examples. The following examples demonstrate several different ways that folders can be opened. Folders must be accessible through the path name supplied in the command.

```
/usr/execfiles/papers
cd /usr/execfiles/papers
```

Either command opens the **papers** folder. It does not matter what folder is open when the command is entered because an absolute path name is used.

```
editor/reviews
cd editor/reviews
```

Either command opens the **reviews** folder. Since a relative path name is used, the folder containing the folder **editor** must be the open folder when the command is entered.

```
cd ..
```

Closes the file or folder.

Starting Programs

There are three ways to start a program file.

1. Move the the file indicator to the desired file (this highlights the file name), then press (Return) (with the command line empty) or press the **Start** ((f1)) function key. (The **Start** ((f1)) function key label appears when a program file is highlighted.)
2. Type the file name into the command line and press (Return). Or you can use the **Echo** ((f2)) key to copy the highlighted file name into the command line and then press (Return).
3. Move the display pointer (with the mouse or the (CTRL) and arrow keys). The location of the display pointer determines the action to be performed when the mouse select button (or (Select) on the User Menu) is pressed. If the display pointer is on the command in the command line, the command is executed as if (Return) had been pressed. If the display pointer is in the folder area of the window, the select button moves the file indicator to that file. If the file indicator is already on the file, the file is started.

Stopping Program Files

Most applications programs for your computer will have an exit procedure, such as pressing a function key labeled Exit or Quit (ususally f8). However, if the program running does not have an exit procedure or the program does not respond to user commands, you may use the keystrokes shown in chapter 2 under "Removing Windows".

Creating Files

The computer creates files automatically when certain commands are executed. There is not a specific command to create a file. When output is directed with the > or >> symbols to a file that does not exist, the system determines that the file does not exist and creates it. The use of these symbols is covered later in this chapter. Many application programs also create files automatically.

Files for storing text are normally created with a text editor, such as *vi* or *MemoMaker*. An editor has its own set of commands that allow you to create files, enter text, and perform editing functions such as moving blocks of text or searching for certain combinations of characters. Files you create with an editor appear as data files in the PAM window and can be moved to other folders, copied, deleted, and manipulated like any other file. (You may need to press Reread (f1) from the PAM screen to display the folder name because not all applications tell PAM the new file exists.) Compiling a data file causes a program file to be created.

Rereading Folders

The Reread (f1) function key is used to display the contents of the open folder. It is possible to make certain changes to the contents of the open folder that do not cause updating of the PAM window. These changes can be a result of running a utility to modify a file, for example. The Reread (f1) function key is displayed on the PAM user menu only when the open folder is highlighted.

You can execute this command whenever you are unsure if an operation has caused a change to the contents of the open folder. Pressing **Reread** (f1) is equivalent to entering *reread* in the command line and pressing Return.

Closing Folders

Pressing the **Close** (f8) function key closes the open folder and displays its outer folder (the folder that contains it) as the open folder.

If the open folder is the root folder (/), then the root folder is redisplayed. Pressing **Close** (f8) is equivalent to entering *close* in the command line and pressing Return.

Creating Folders

The *makefolder* command is used to create folders. The folders are created in the open folder if the folder name is a simple name. If a path name is supplied, then the folder is created in the specified folder. The following command is used for creating one or more folders:

`makefolder` *folder_name ...*

Duplicate file or folder names in the same folder are not allowed. If you attempt to create a folder with a duplicate name, a message is displayed. Usually, the last character of the folder name can be changed or a character can be added to make the name unique. Also, folder names are limited to 14 characters and should not include any of the special characters (described in the *Integral Personal Computer User's Guide*).

Path names can be used to specify the creation of a folder in an existing folder that is not currently open. The last folder name in the path name is used for the new folder.

Examples. The following examples create folders at different levels of the file system.

```
makefolder memos reviews
```

Creates the folders **memos** and **reviews** in the open folder.

```
makefolder /usr/execfiles/papers
```

Creates a folder named **papers** in the **execfiles** folder. The folders **usr** and **execfiles** must already exist.

```
makefolder editor/reviews
```

Creates a folder named **reviews** in the existing folder **editor**, which is contained in the open folder.

Deleting Files and Folders

The *delete* command is used to delete files or folders. The file to delete can be the simple name of a folder or file in the open folder. A path name is used for deleting files or folders contained in other folders. Several files or folders can be deleted with the following commands:

```
delete file_name ...
delete folder_name ...
```

Be careful when using the *delete* command. It can be used to delete any file or folder in the file system, with the following exceptions:

- Files or folders stored on a disc that is write-protected.
- Files or folders that do not have write permission.
- Files or folders whose outer folder does not have write permission.

- Files or folders that are currently being accessed or viewed (with the *view* command).
- The open folder.
- Folders that are not empty. All files or folders stored in a folder must be deleted before the folder itself can be deleted.
- The outermost folder on a disc.

Do not attempt to delete files or folders created for use by the system. (For example, **rom** and **dev**.) It is possible to delete files and folders that are necessary to the functioning of the computer. The folders **dev** and **tmp** are examples of system folders stored in the root folder (**/**). If you delete these folders, or any of the files they contain, the computer will fail to operate properly when certain operations are performed. System files and folders are recreated from internal storage at each power on by the operating system. If you accidentally delete one of these files or folders, you should turn the computer off then on again to restart the operating system.

If a folder appears empty in the PAM window but a message is displayed that it is not empty when you attempt to delete it, it probably contains a hidden file or hidden folder (a folder that has a name beginning with "."). The folder cannot be deleted until all the files and folders it contains are deleted (with the exception of the "." and ".." files created by the system—the user does not need to delete these files from a folder in order to delete the folder). To list the complete contents of any folder and obtain the names of any files or folders that are hidden in it, use the **ls -a** command. (Refer to chapter 4 for a description of the **ls** command.) After deleting these files or folders, the folder they were stored in can be deleted.

Examples. The following commands are examples of how you can delete files and folders.

```
delete old_product
```

Deletes a file called **old product** from the open folder.

```
delete /usr/editor/text1
```

Deletes the file **text1** from the folder **editor.** The command can be executed regardless of which folder is open.

```
delete reviews
```

Deletes the folder **reviews** from the open folder.

The Delete (f7) function key writes "delete" followed by the highlighted file name in the command line. To execute the command, you then press Return. The Delete (f7) function key is a typing aid for deleting files and folders in the open folder.

Moving Files

The Move (f4) function key can be used to move a file stored in one folder to another folder. To use *move*, first highlight the file you wish to move and then press the Move (f4) key. You will be prompted to supply the destination file name. The *move* command can also be entered in the PAM command line.

```
move file_name
move file_name ... folder_name
move file_name1 to file_name2
```

The file names identify the files to be moved. The folder name specifies the folder the files are moved to. If you only enter a file name, then you are prompted to supply the folder name with the *to* command. It is necessary to use a path name for either the file name or the folder name unless both of these are stored in the open folder. If you specify two file names, the command moves the first file to the second file. (Full pathnames for the files can be used.) Note that specifying an existing file as the second or last parameter of a move command causes its contents to be overwritten (if write permission exists on the file) with the file being moved.

Examples. The following commands show how files can be moved to different folders when it is not necessary to exchange discs.

```
move productA
```

Begins an operation to move the file **productA** from the open folder to another folder.

```
to /usr/editor/memos
```

Completes the move operation by moving **productA** from temporary storage to the **memos** folder.

Folders can also be moved to a different folder on the same disc. The command has the forms:

```
move folder_name
move folder_name1 ... folder_name2
```

Moving Files Between Discs

If a file to be moved and the folder it is to be moved to are not on the same disc and cannot be accessed through the file system without changing discs, then it is necessary to use a temporary storage area while the discs are exchanged.

To do this, first, move the files from the disc they are currently stored on to the root folder with the following command.

```
move file_name ... /
```

After the command completes, insert the disc containing the folder you want to move the files to. If the full path name of the destination folder is *folder_name* and the open folder is **/**:

```
move file_name ... folder_name
```

Examples. The following examples show how the **move** command can be used to move files from one folder to another when both folders are not on the same disc.

```
move productA productB /
```

Moves the files **productA** and **productB** from the open folder to the root folder. Exchange the discs after this command completes.

```
move productA productB /newplace
```

Moves the files **productA** and **productB** from the root folder to the open folder **newplace**.

Renaming Files and Folders

The Rename (f6) function key can be used to change the name of a file or folder. It can change the name of any folder or file in the file system except the name of the root folder of a disc and **rom**. The root folder name (of a disc) corresponds to the disc name and is changed with the *rename_disc* utility. To use Rename (f6), highlight the file or folder whose name you wish to change and press the Rename (f6) function key. You will be prompted to enter the new name. This command can also be entered in the command line in the following forms:

```
rename file_name
rename folder_name
rename file_name1 file_name2
rename folder_name1 folder_name2
```

The file or folder to be renamed is the first parameter. If a second parameter is included, it is used as the new name. Otherwise, you are prompted to enter the new name. If the file or folder to be renamed is not in the open folder, then a path name can be used to identify it. The new name entered should include a similar path name.

Examples. The following examples show how *rename* is used to change the names of files and folders.

```
rename Book1
```

Begins a rename operation to change the name of the file **Book1** in the open folder.

```
to Chapter1
```

Completes previous rename operation by specifying **Chapter1** as the new name of the file.

```
rename productA productA1
```

Renames the file **productA** in the open folder to **productA1**.

```
rename /usr/editor /usr/docfiles
```

Renames the folder **editor** to **docfiles**. The command can be executed from any open folder, since the full path names were used.

```
rename papers/Book1 papers/Thesis
```

Renames the file **Book1** in the **papers** folder to **Thesis.** The open folder must contain the folder **papers.**

Copying Files

The Copy (f5) function key can be used to make a duplicate file in the same folder or another folder. Backup files can be created by making duplicates of a file, for example. To use Copy (f5), highlight the name of the file you wish to duplicate and press Copy (f5). You will be prompted to enter the name of the folder to which you want the file copied.

The *copy* command may be entered in the command line as follows:

```
copy file_name
copy file_name1 file_name2
copy file_name folder_name
copy file_name ... folder_name
```

If you enter one parameter with the copy command, the copy operation is started and you are prompted to complete the operation with the *to* command.

When two file names are used as parameters, a name must be specified for the copy of the file. A path name can be used to specify that the file is to be copied to a different folder. If you specify a name for the copy of the file and a file with that name already exists in the specified folder, then the copy operation overwrites the file with the new name (unless that file is write-protected.

If the last parameter in the *copy* command is a folder, then the specified files are copied to it. The names of the files are not changed. The last parameter specifies the folder that the copies of the files are placed in.

Examples. The following examples show how files can be copied using different forms of the *copy* command when exchanging discs is not necessary.

```
copy Book1
```

Begins an operation to copy the file **Book1** from the open folder to the specifed file or folder.

```
to Chapter1
```

Completes the previous operation by copying the file **Book1** to the file **Chapter1.**

```
copy Book1 ..
```

Copies the file **Book1** to the folder containing the open folder.

```
copy memos/meeting1 memos/minutes1
```

Copies the file **meeting1** to **minutes1.** The open folder must contain the folder **memos.**

```
copy productA productB /apps
```

Copies the files **productA** and **productB** to the **apps** folder.

Copying Files Between Discs

Files can be copied between two discs with the *copy* command by temporarily storing a copy of the file while discs are exchanged. Two methods can be used for copying files between discs. The first method allows you to copy one file to a different disc, the second method allows multiple files to be copied.

To copy one file to a different disc, the *copy* command is entered with one parameter. This is equivalent to highlighting a file in the open folder and pressing Copy (f5).

```
copy file_name
```

After the file is copied to the **tmp** folder for temporary storage, you are prompted to use the *to* command to complete the operation. Insert the disc containing the folder the file is to be copied to and enter one of the following commands:

```
to file_name
to folder_name
```

The second method can be used to copy multiple files between discs using the root folder as a temporary storage area. After the files are copied to the root folder, the disc they were originally stored on can be removed. To copy the files to a temporary storage area, type:

```
copy file_name ... /
```

Next, insert the disc the files are to be stored on. The files can be moved from the root folder to a folder on this disc with the *move* command, as follows:

```
move file_name ... folder_name
```

Examples. The following examples show how copies of files can be created on separate discs when both discs cannot be included in the file system at the same time.

```
copy Chapter1
```

Begins a copy operation for copying the file **Chapter1.** When you are prompted to enter the *to* command, exchange the discs.

```
to Chapter1X
```

Completes the previous copy operation by copying the file **Chapter1** to the file **Chapter1X.**

The following two commands can be used to copy multiple files between discs:

```
copy productA productB /
```

Copies the files **productA** and **productB** from the open folder to the root folder. Exchange discs after this command completes.

```
move /productA /productB /back
```

Moves the copies of the files **productA** and **productB** from the root folder to the **back** folder.

Copying and Moving Files to the Electronic Disc

As you have just read, files may be copied to the root folder stored on the electronic disc for temporary storage while discs are exchanged. You can also use the *copy* or *move* commands to copy or move files to the electronic disc to reduce the number of input and output operations to a disc drive. Files may be stored in the root folder or you can use the *makefolder* command to make a new folder in the root folder. Remember that files stored in the electronic disc are erased when the computer is turned off.

Accessing Files with `view`

The View (f1) function key can be used to display the contents of any file in an alpha window. To use this command, press View (f1) to display the contents of the highlighted file. This command can also be entered in the command line as:

`view` *file_name*

When a file is accessed with the *view* command, it is displayed one page at a time. The window the file is displayed in provides two menu labels, More (f1) and Quit (f8). Pressing More (f1) displays the next page of the file. The space bar has the same function as More (f1) when a *view* window is active. Pressing Quit (f8) ends viewing of the file and removes the window.

The *view* command should be used to display data files or script files. Program files (that are not script files) and other files that do not contain text should not be displayed with *view*. If the file accessed does not contain readable text, the data it contains will be interpreted as control characters and other non-displayable characters.

To view a data file, you can:

- Use the file indicator to select a file, then press the View (f1) function key.
- Enter the name of the data file on the command line and press Return.
- Enter `view` *file_name* in the command line and press Return.

Examples. The use of *view* to display the contents of a file is shown in the following examples.

The viewing window has the same name as the file to be viewed except when the *view* command is explicitly used.

```
view /usr/execfiles/papers/Book1
```

Displays the file **Book1** in a *view* window. The command can be executed from any folder since an absolute path name is used.

```
view productA
```

Displays the file **productA** for viewing. This file must be in the open folder.

```
view apps/file/workfile
```

Displays the file **workfile.** The open folder must contain the folders **apps** and **file**.

Printing Files

The *print* command is used to print files on the active printer device. Unless the active printer has been changed with the *printer_is* utility, the active printer is the internal printer. Any number of files can be printed with the following command:

```
print file_name ...
```

If several file names are entered, then one file starts printing and the additional files are placed in a *waiting list* for the active printer. This waiting list is referred to as a *printer queue*. When the file being printed completes printing, another file in the printer waiting list begins printing. When several file names are specified with a single *print* command, the files are printed in the order that they are listed in the command line. If separate *print* commands are used, the order of printing is not necessarily the order in which the statements were executed. The printer continues to operate until the printer queue is empty.

Each file to be printed must remain accessible to the system until it is finished printing. If all the discs containing the files to be printed cannot be included in the file system at the same time, then the root folder can be used for temporarily storing the files. Use the *copy* command to place a copy of each file to be printed in the root folder before executing the *print* command. After printing is completed, you can remove these files from the root folder with *delete*.

Only files containing readable text should be printed. The printing of program files or other files that do not contain readable text can result in control characters and other non-printable characters being output.

Examples. The following commands cause data files to be printed on the active printer.

```
print minutes1
```

Prints the file **minutes1** from the open folder.

```
print /productA /productB /productC
```

Prints the files **productA, productB,** and **productC** from the root folder.

```
print execfiles/papers/Book1
```

Prints the file **book1.** The open folder needs to contain the **execfiles** folder when the command is executed.

If, for some reason, you wish to stop a file from printing, the *stopprint* command can be used to stop the printing. This command cancels printing of the currently printing file and moves the paper to the next top-of-form position. If another file remains in the printer waiting list, it begins printing. Files cannot be removed from the printer waiting list with *stopprint.* The command is entered as follows:

```
stopprint
```

Sending Text

Text can be sent within the system from the PAM window to a file, device, or window. Two commands are used for sending text: *send* and *echo.* Entering the *send* command or pressing Send (f3) sends (or writes) the full path name of the highighlighted file to the next window down. After this operaton, the next window down becomes the active window.

Echo

Pressing the Echo (f2) function key copies the highlighted file or folder name to the command line. The *echo* command copies a string to a specified file, window, or device. The string and the file or window it is to be copied to are indicated as follows:

```
echo string
echo string >file_name
echo string >>file_name
echo string #window_name
```

The string can be copied to a file with the > or >> symbols or can be copied to a window with the # symbol. If the window specified does not exist, it is created. If no destination is specified, the string is printed in a separate *echo* window.

Examples. The following examples show how the *echo* command can be used to send text and commands within the system.

```
echo Have a nice day. >/dev/lp
```

Copies the text `Have a nice day.` to the active printer. (Refer to chapter 4 for a description of the *printer_is* utility.

```
echo It is raining today.
>>/usr/execfiles/diary
```

Adds the text `It is raining today.` to the end of the data file `diary`.

```
echo This is a window. #wind1
```

Copies "This is a window." to the window **wind1**.

Redirecting Input and Output

The input and output of programs can be redirected by including certain symbols in the command line when a program is executed. The redirection only affects the input and output of the program it is specified for and is canceled after the program has completed executing. The same symbols that are used to redirect the input and output of programs can also be used with utilities and commands built into PAM when they are executed from the command line.

Redirecting input and output is a convenient way of selecting what files or devices a program uses when it is executed. The output of a program that is normally displayed can be sent to the active printer by including a symbol and file name in the command line. This does not affect the functioning of the program because the destination of output from the program is changed at the system level.

Input redirection is not used as frequently as output redirection because the keyboard is the default for program input and is convenient to use. If the keyboard cannot be used for providing program input, then a file to be used for program input can be specified. The file is designated by including a symbol in the command line when the program is executed.

Every program that is run has three data paths associated with it. These are standard input, standard output, and error output. Programs use these data paths to interact with you. Standard output and error output are directed to the window a program is running in. When this window is the active window, the keyboard can be used for the standard input of the program.

The symbols for redirecting input and output allow you to change a specific data path of a program while leaving its other data paths unchanged. For example, you can specify a different device for standard output. Using the device file `/dev/lp` for standard output causes output from a specific program to be printed on the active printer.

Another alternative for redirecting input and output is designating a window for a program to run in. This window is then used for the standard input, standard output, and error output of the program. A window is created by the PAM program when a symbol is included in the command line that identifies the window name to use for the program.

Symbols to redirect the input or output of a program or specify a window for it to run in are entered in the command line when the program is started. When Return is pressed, the program begins executing with the data paths specified by the redirection symbols. To specify redirection of input or output for a program, each file or window name is preceded by a redirection symbol, as follows:

program_name symbol file_name
program_name symbol file_name ...
program_name symbol window_name

The program and any parameters it requires are entered first. Then changes to program input or output are specified. Each redirection symbol designates a change to a data path or provides a window name. Spaces between the redirection symbols and the files or window names are optional. The symbol identifies the name that follows it as a file for input or output or the name of a window. The redirection symbols are listed in the following table:

Redirection Symbols

Symbol	Function	Example
<	Read standard input from an existing file.	`program1 <input.data`
>	Write standard output to a file.	`program2 >output.data`
>>	Append standard output to an existing file.	`sample.prog >>output.data`
^	Write error output to a file.	`ABC ^error.listing`
^^	Append error output to an existing file.	`pgm.tst ^^error_log`
#	Use the specified window for standard input, standard output, and error output.	`newprog #test`

Information in files or windows will be lost when certain redirection symbols are used. If output from a program is directed to an existing file with the > or ^ symbols, the file will be emptied and overwritten when the program starts.

If a file you specify with a redirection symbol is not in the open folder, then you should use a path name to identify it. The following actions are taken when the system does not locate files specified with redirection symbols:

- If a file specified for input with the < symbol is not located, then an error message is displayed.
- If a file specified for output with the > or ^ symbols is not located, then it is created and used for program output.
- If a file for appending output to with the >> or ^^ symbols is not located, then it is created and used for program output.

Be careful not to specify the same file for standard input and standard output or for standard input and error output. When input and output operations access the same file, the results are unpredictable.

Examples. The following examples show how the data paths of programs, commands, or utilities can be modified with the redirection symbols.

```
prog1 ^errlist
```

Runs the program *prog1* and sends errors to the file *errlist.*

```
CHItest <data1
```

Runs the program *CHItest* using the file *data1* as input.

```
ABC >data1 ^err1
```

Runs the program *ABC* using the file **data1** for standard output. The file **err1** is used for error output.

```
Chapter1 >/dev/lp
```

Prints the file **Chapter1** on the active printer.

```
date >>syslog
```

Adds the current time and date to the end of the file **syslog.**

```
newprog #test
```

Runs the program *newprog* in a window named **test.**

```
view doc1 #manual
```

Displays the file **doc1** in a window named **manual.**

```
yodes >/dev/screen/list
```

Runs a program named *yodes* and displays the standard output in the existing **list** window.

```
brkdncng ^/dev/screen/errs
```

Displays error output from *brkdncng* in the existing window named **errs.**

Connecting Programs with Pipes

Two or more programs can be connected so that the output of one program is used as the input of another program. The data path that joins the programs is called a *pipe*. Pipes allow you to redirect program input and output without the use of temporary files.

When programs are connected with pipes, the operating system synchronizes the input and output between the programs. The programs are executed concurrently. When a pipe is full, the program outputting the data to it is halted. When a pipe is empty, the program inputting data from it is halted.

The data paths of the programs that are connected with the pipes are standard input and standard output. The pipes only transfer data in one direction, from the standard output of one program to the standard input of another program. The window the programs are running in is used for standard input to the first program in the pipe and for both the standard output and error output from the last program in the pipe.

The vertical bar (|) is the "pipe" symbol. Parameters for the programs are listed after the program name, as usual, but before the | symbol. Spacing between the program names and vertical bars is optional. The syntax used for connecting programs with pipes is as follows:

program_name | *program_name* | *program_name...*

The following command could be used to specify the concurrent execution of three programs with pipes, as shown in the preceding illustration.

```
program_A | program_B | program_C
```

Pipes operate on or transform data by separate programs in stages. For example, *program_A* could obtain data from a user. This data would be passed by the first pipe to *program_B* where it would be checked for validity and processed. The processed data could then be passed by the second pipe to *program_C* for formatting into a report.

The redirection symbols described in the last section can be used for programs connected with pipes. However, only the data paths that are not connected with pipes can be changed. If you specify a change to a data path being used with a pipe, then an error occurs. The following changes are permitted:

- The standard input of the first program in the pipe can be redirected with the < symbol.
- The standard output of the last program in the pipe can be redirected by using the > symbol or appended to an existing file with the >> symbol.
- The error output for any program in the pipe can be redirected with the ^ symbol or appended to an existing file with ^^.

Examples. The following commands show how programs can be connected with pipes and how additional changes can be made to data paths with redirection symbols.

```
test_prog1 | /usr/output_prog
```

Connects the programs *test_prog1* and *output_prog* with a pipe.

```
program.A <data.A ^errors.A|program.B ^errors.B
```

Connects the programs *program.A* and *program.B* with a pipe and makes additional changes to the standard input and error output paths. For *program.A,* the file **data.A** is used as standard input and **errors.A** is used for error output. The file **errors.B** is used for error output from *program.B.*

```
get_it|check_it|process_it|format_it >/dev/lp
```

Runs four programs connected with pipes and prints the output of the fourth program on the active printer.

Specifying Sequential Processing

When a program begins, it normally starts executing immediately. New programs can be run whether or not the last program started has finished executing. In some cases, it is useful to be able to prevent additional programs from being run until a specific program has completed. This is called *sequential processing.* When a program is run with sequential processing, additional programs or commands cannot be entered from the PAM window until the specific program has finished executing. Sequential processing can be specified for any file or command; although, it is normally used only for programs.

Sequential processing can be specified for individual commands you enter in the command line. Placing a semicolon (;) after the last character in a command specifies sequential processing for that command. For example, to run the program *backup1* with sequential processing, use the following command:

```
backup1 ;
```

If parameters are required by the program, they are entered as usual. The semicolon is placed after the last parameter or redirection symbol, as in the following command:

```
test_program 4 5 >test_output ;
```

While a program is running as a sequential process, PAM places any commands or other keystrokes you enter (while the PAM window is active) in a buffer. There is no response to keyboard activity with the PAM window until after the program has completed. The characters in the buffer for the PAM window are displayed in the command line after the program completes executing.

Programs that are already in progress when a program with sequential processing is executed continue to run as usual. While a program is running as a sequential process, you have the option of waiting for the program to finish or using applications already running in other windows. Other windows are selected by pressing (Shift)(Select).

Examples. The following commands specify sequential processing for programs.

```
cleanup_prog ;
```

Runs the program *cleanup_prog* as a sequential process.

```
/usr/otherapps/backup1 #test_window ;
```

Runs the program *backup1* as a sequential process in a window called **test_window.**

```
text_formatter execfiles/papers/Book1 ;
```

Runs the program *text_formatter* as a sequential process. The data file **execfiles/papers/Book1** is a parameter for this program.

```
test1_diag ; test2_diag;
```

Runs the program *test1_diag,* waits for it to finish and then runs **test2_diag**.

Several programs can be run sequentially using the following syntax:

program_name; *program_name*; . . .

After the first program is run and completed, the second program begins. After that program ends, the next program begins, and so on. All programs run this way are executed in the same window. The window has the name of the first program.

Specifying Non–Sequential Processing

Programs can also be run non-sequentially (that is, each program runs without waiting for the previous program to complete). Following the program name with & specifies non-sequential processing. The syntax is:

program_name& *program_name1*& *program name2*&...

To run a single program non-sequentially, the & is optional. (That is, the default processing of programs is non-sequential.) If more than one program is specified in a command line, the window the programs run in has the name of the first program.

Command Line Quoting

To prevent PAM from processing various special characters, you can enclose those characters in quotation marks. PAM recognizes the following quotation marks: " (double—quote) and ' (single—quote).

When a group of one or more characters are enclosed within a set of either single— or double—quote marks, PAM does not interpret them as special characters. You can also use \ (backslash) to quote a single character. The character following the backslash is not interpreted by PAM as a special character.

Examples. Either two double—quote marks or two single—quote marks can be used. But you can't use a single— and a double—quote mark together.

```
echo "some special characters: # < >"
echo 'some more special characters:  & ; '
echo here is a special character \;
```

Quotes can also be quoted, but they must be enclosed within a pair of different quote marks. That is, one or more single quotes can be enclosed within a pair of double quotes, and one or more double quotes can be enclosed within a pair of single quotes. For example:

```
echo 'here are some double quotes " " '
echo "here are some single quotes ' ' "
```

Specifying Operations with Files

The remainder of this chapter covers the use of three types of files that you can create to enhance the operation of the computer. These are script files, **Autost** files, and environment files. The following information should help you determine whether using these files would help you use the computer more efficiently.

- A script file is a type of program file that contains a series of commands you can enter in the command line. Executing a script file allows you to execute all the commands it contains by entering only the script file name. A script file can be used at any time.
- An **Autost** file is used to perform desired operations at the beginning of a session. The file must be named **Autost** and be stored on the outermost folder of the disc in the internal disc drive for the system to locate it at power on. **Autost** files are often used to start an application running so that no initial choices need to be made with the PAM window.
- An environment file is a data file that contains certain variable assignments for setting the active environment. The system accesses the disc in the internal drive at power on to set the active environment with a file named **.environ.** (The default environment file is **/rom/.environ**.) The active environment may include a search path for locating files, a home folder that can be easily accessed, and other information about the computer for application programs.

Script Files

Script files contain commands and can be executed like a program. By creating a script file, you can design commands for your own needs. When the name of the script file is entered in the PAM command line, the commands in the file are executed as if you entered each one individually. Any command that can be entered in the command line can be placed in a script file. The names of other script files can be placed in a script file and are used as subprograms or "subscripts."

Depending on the commands you include in a script file, command output may or may not be displayed when you execute it. Output from PAM commands, such as *move* or *copy* (except error messages), is suppressed when the commands are executed from a script file. Output from disc-based utilities and other programs is displayed as if you had executed them from the command line.

Script files are first written as data files with a text editor. However, before a data file containing commands can be executed as a script file, its execute permission must be changed so it is identified as a program file. The *chmod* command can be used to change file permissions. (Complete information about the *chmod* command is provided in chapter 4.)

With script files, sequential processing is the default for all commands. The command on the second line of the script file is not started, for example, until the command on the first line is finished executing. Multiple commands can be run simultaneously from a script file by entering the ampersand (&) character after each command that is not to be processed sequentially. Placing an ampersand after a command specifies that the system does not need to wait for it to finish executing before executing the next command in the script file. Using an ampersand after each command in a script file runs all the commands as if they were entered in the command line. Omitting an ampersand after a command in a script file is equivalent to placing a semicolon after the same command when it is entered in the command line.

Parameters needed for commands in a script file are entered in the command line when the script file is executed. For example, to execute a script file called **print_it** that requires one file name **lst**, as a parameter, the command would be the following:

```
print_it lst
```

Lets assume for this example that the file **print_it** has the following contents:

```
copy $1 /
print /$1
delete /$1
```

In the **print_it** script file, **lst** is assigned to the variable `$1`. First, this file is copied to the file **lst** in the root folder. Then the file **/lst** is printed on the active printer. Finally, **lst** is deleted from the root folder. Because sequential processing is the default in script files, the *print* command is not executed until the file is copied to the root folder and the file is not deleted until printing is complete.

The parameters for script files are called positional because they are assigned to variables in the script file in the order they are entered in the command line. The parameter values you enter are passed to the variables `$1`, `$2`, and so on up to `$9`, in the script file. Multiple parameters must be separated by one or more spaces. The first parameter is assigned to the variable `$1`, the second parameter is assigned to `$2`, and so on. The null string is assigned to variables for parameters you do not enter. Using the variable `$0`, the name of the script file can be used within the script.

When all variables are referenced at once or over nine variables are needed in a script file, the variable `$*` can be used. All the parameters entered in the command line are passed to the variable `$*` as a long character string. This variable is also useful when a different interpretation of parameters, such as character string values, is necessary for the commands in a script file.

Autost Files

The use of an **Autost** file enables you to have a file execute automatically at power on. The file type determines what action is taken. If the **Autost** file is a data file, it is displayed. If it is a program or a script file, it is executed. The system checks the disc in the internal drive for a file with the name **Autost** at power on. If a file named **Autost** is stored in the uppermost folder of this disc, it is executed. PAM will accept command input after the file is completed. If the system does not locate an **Autost** file, the PAM window is displayed.

Any file can be used as an Autost file by using the *rename* command to change its name to Autost. Names such as autost or AUTOST should not be used because the system differentiates between lowercase and uppercase letters. Unless the file name is Autost, the file will not be accessed at power on.

Environment Files

Environment files provide information to the system for defining the active environment. The active environment is a collection of settings that the system adds to programs and commands when they are executed. These settings provide information about your system and the files and folders it contains. The computer system uses an environment file called **.environ** for defining the active environment at power on.

When the computer is switched on, the system checks the uppermost folder of the disc in the internal disc drive for a file called **.environ.** If such a file is present, it is used to define the active environment for the current session. If an **.environ** file is not present, the default environment file stored in the computer is used. Different active environments can be selected during a session by specifying an environment file with the *getenv* command. The uses of an **.environ** file include the following:

- The system uses the **.environ** file as a source of information when conducting searches. When the name of a program is entered as a command, the PATH variable provides a list of folders to search when the program is not in the open folder.
- Utilities can determine information about the system environment by reading variables in the **.environ** file. The *time_of_day* utility, for example, accesses the variable TZ to determine what time zone the computer is in.
- Commands can use information in the **.environ** file as a supplementary source of information. The *cd* command uses the HOME variable to determine what folder to open when it is executed without a parameter, for example.
- Application programs utilize the **.environ** file to obtain information about how various features are implemented on the computer.

An editor is normally used for making changes to the **.environ** file. Changes can be made to the assignment strings of existing variables. Or, different variables can be added to the file as required by your applications. The following form is used for all variable assignments:

variable=string

In addition,

- Blanks within a variable assignment terminate that assignment. For example, TERM= sets the terminal type to the null string. The end-of-line indicator (usually the carraige return character) should be used to terminate the variable assignment.
- Colons are used to separate parts of a variable assignment. With the PATH variable, colons separate the folder names.

Changing the **.environ** file does not cause a change to be made in the active environment until the computer is switched on and off again. The active environment is set from the **.environ** file stored on the disc in the internal drive at power on. The *getenv* command allows you to change the environment during a session without turning off the computer. Changes you can make to the following variables are described in the next few sections:

- The HOME variable sets the home folder. Executing the *cd* command without a folder name causes this folder to be displayed.
- The PATH variable sets the search path for programs and script files. When a program name is entered in the command line that is not stored in the open folder, the folders included in the search path are searched in order until the the specified file is found.
- The TERM variable provides the terminal type. This can be accessed by application programs to determine whether level 0 or level 1 features are implemented, for example.
- The TERMCAP variable provides information about terminal windows with a terminal capability data base. This data base can be used by applications to identify what capabilities are supported on the computer and how they are accessed.
- The TZ variable provides the time zone information. This can be changed so that the `time_of_day` utility displays the correct time zone.
- The ACTION variable specifies a program to execute when a data file name is entered as a command.

Setting the Home Folder. The home folder is the folder accessed by executing the **cd** command without a parameter. It is convenient to set a folder you frequently access as the home folder. If the folder currently being used as the home folder is not the one you use most often, then the HOME variable in the **.environ** file can be changed to another folder. The folder name assigned to the HOME variable should be an absolute path name.

```
HOME=folder_name
```

An example of setting the home folder is changing it from **usr** to a folder called **/usr/execfiles**. The folder you specify must already exist. To change the home folder, access the **.environ** file with an text editor and change the HOME variable to the following:

```
HOME=/usr/execfiles
```

Although the value of the HOME variable has been changed, the home folder used by the system will be **usr** until the active environment is set again at power on or it is changed with the *getenv* command.

Setting the Search Path. The search path is a list of folders that is searched for programs in a specific order. When a program name is entered in the command line and the program file is not in the open folder, the system checks the folders included in the search path for the specified file. The order of folders in the search path determines the efficiency of the search and, when duplicate file names exist, it determines which file is accessed.

PAM uses search paths only for program file names. Files used as command parameters are not located with search paths.

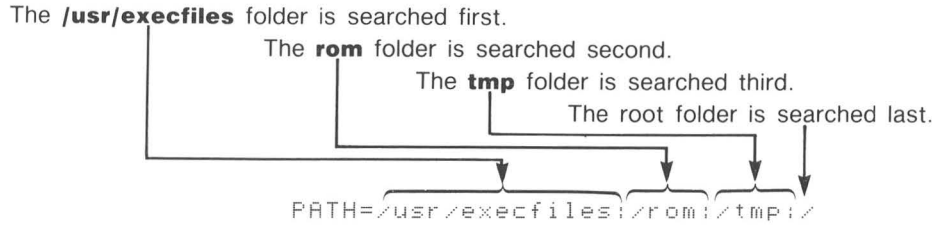
The system searches for files starting with the leftmost folder in the search path. If a file is not found, the next folder is searched. When a match is found, the search ceases, and the file is accessed. If no match is found, an error message is displayed.

The path names of the folders in the search path are separated with colons, as follows:

PATH=*folder_name* : *folder_name* ...

The order that folders are searched when a command is entered in the command line is shown in the following illustration. If a file in the open folder has the same name as the command, it is accessed and a search is not performed. If the search is unsuccessful after searching the last folder in the search path, then an error message is displayed.

Sample Search Path


The /usr/execfiles folder is searched first.
The rom folder is searched second.
The tmp folder is searched third.
The root folder is searched last.
PATH=/usr/execfiles:/rom:/tmp:/

If the root folder is frequently used for storing programs and files, the ordering of the folders in the search path can be changed to improve search efficiency. The following assignment changes the sample search path so that the root folder, represented with /, is searched first instead of last:

```
PATH=/:/usr/execfiles:/rom:/tmp:
```

Another change that can be made to a search path is adding another folder to it. If a folder named **/usr/solutions** is used for storing frequently executed programs, then it can be included in the search path. This would allow programs stored in **/usr/solutions** to be executed from any open folder without specifying a path name. Note that many application programs do not use the PATH variable and can only locate files outside of the open folder when a path name is used. The following variable assignment adds the **/usr/solutions** folder to the sample search path:

```
PATH=/usr/execfiles:/rom:/tmp:/:/usr/
 solutions:
```

If a folder is specified for the PATH variable that is not currently included in the file system, then it is ignored. An error message is displayed only when programs entered in the command line cannot be located.

Setting the Action. The computer takes certain default actions when files are selected depending on file type. The variable ACTION can be added to the **.environ** file to change the action taken when data files are selected. If the ACTION variable is not present, data files are displayed when they are accessed. When the variable ACTION is included in the **.environ** file and is set equal to a program name, then selecting a data file causes that program to begin executing.

If you change the default action for accessing data files by including the ACTION variable in your **.environ** files, the view command can still be used to display data files.

Assigning a program to the ACTION variable enables application programs to be automatically executed whenever the name of a data file is entered in the command line. The name of the data data file selected is passed to the program as the first positional parameter so that entering the name of a data file then becomes equivalent to entering the program name followed by the data file name in the command line. A program is identified as the default action as follows:

ACTION=*program_name*

To specify that the program *MemoMaker* be executed, for example, add the following line to the **.environ** file:

```
ACTION=MemoMaker
```

If a simple name is used to identify the program, then the folder it is stored in should be included in search path.

Selecting the Environment with getenv

The active environment set at power—on can be changed by selecting another environment file with the *getenv* command. The *getenv* command replaces the active environment with the contents of the specified file. You specify the environment file for *getenv* to access by entering it as a parameter, as follows:

```
getenv file_name
```

To change the active environment to the settings specified in a file named **sysenv,** the command is:

```
getenv sysenv
```

The *getenv* command is convenient for changing the environment after editing the **.environ** file without restarting the computer. The environment can be changed by executing *getenv* from a script file, but the changes apply only to the commands executed from the script file.

4 Command and Program Dictionary

Introduction

The commands described in this chapter are built into the computer and are available whenever you're using PAM. The programs described in this chapter reside on the following discs:

- The Utilities disc.
- The Standard Applications disc.
- The HP-UX Commands disc.
- The System disc.
- The Diagnostic disc.

In order to use a specific program, be sure that the computer can access the appropriate disc. You can either:

- Open the folder containing the program, highlight the program name in the PAM window, and start the program, *or*
- Type the *simple name* or *path name* of the program, followed by *parameters* (if any), and press (Return).

If a program requires one or more parameters, such as a *disc_name* or a *window_name*, then you should include the parameters in the command line, separated by one or more *spaces* from the name of the program and from each other.

Example. To use the *plotter_is* program and the PAM window to create a graphics window named `WaterPressure`, you would:

1. Insert the Utilities disc in the internal disc drive.
2. Ensure that PAM shows `/util`—the name of the Utilities disc—as the open folder.
3. Highlight the name of the desired program—`plotter_is`—in the PAM window.
4. Press Echo ((f2)) to "type" the *plotter_is* program name (with a trailing space) in the PAM command line.
5. Type the desired parameter—`WaterPressure`—after the program name so that the command line reads:

   ```
   plotter_is WaterPressure
   ```

6. Press (Return) to start the program.
7. Press (CTRL)(Select) to show the new graphics window. (The graphics window will remain unseen until you select it with (CTRL)(Select) or until the window receives program output.)

PAM is sensitive to differences between uppercase and lowercase letters. For example, PAM will assume that a program named "RUNME" is different from a program named "runme". Be sure that you type commands in the PAM window with the same case and spacing as that shown in the examples.

If a program creates a window during its execution, you can eliminate the window when you've finished using the program:

1. Select the program window using (Shift)(Select), (CTRL)(Select), or the mouse.
2. Press **Exit** ((f8), if available) or (Stop) (at the upper left of the keyboard).

Parameters

Following are descriptions of the *parameters* for the commands and programs in this chapter.

Command and Program Parameters

Parameter	Description	Examples
drive_name	The letter **D** followed by three digits, indicating *volume_number*, *HP-IB_address*, and *unit_number*. (Refer also to Chapter 5, "Installing and Using Peripherals.")	`unmount_disc` *drive_name* `umount_disc D140`
disc_name	One to six characters, which may include letters, digits, and the following symbols after the initial character: period (.), hyphen (-), and underscore (_).*	`format_disc` *disc_name* `format_disc JAWS`
file_name†	One to fourteen characters, which may include letters, digits, and the following symbols after the initial character: period (.), hyphen (-), and underscore (_).* (A *file_name* beginning with a period will not be visible in the PAM window.)	`rename` *source_file_name* *destination_file_name* `rename OldFile NewFile`
folder_name	Same syntax as that for a *file_name.** (A *folder_name* beginning with a period will not be visible in the PAM window.)	`makefolder` *folder_name* `makefolder Brand-New`
HP-IB_address	One or two digits, from **0** through **30**, that uniquely identifies a device connected to the internal HP-IB interface.	`printer_is` *HP-IB_address* `printer_is 2`

* Some programs allow a wider range of special symbols in names than the period (.), hyphen (-), and underscore (_). However, you should avoid using the following symbols in names because they are reserved for other uses: slash (/), backslash (\), vertical bar (|), greater-than symbol (>), less-than symbol (<), semicolon (;), number sign (#), circumflex (^), ampersand (&), asterisk (*), apostrophe ('), double quotation mark ("), equals sign (=), question mark (?), opening parenthesis ((), closing parenthesis ()), opening bracket ([), and closing bracket (]).

† Wherever a *file_name* appears in this chapter, you may instead use a relative or absolute *path_name*.

Command and Program Parameters (continued)

Parameter	Description	Examples
program_number	An integer greater than 1 that specifies a program loaded in system memory.	`terminate` *program_number* `terminate 19`
RS-232_device_name	The device name that uniquely identifies a serial interface connected to the computer. Consists of the letters `cul` and a digit string from `00` *through* `09`.	`printer_is` *RS-232_device_name* `printer_is cul00`
window_name	Same syntax as that for a *file_name.** Precede with a number sign (#) *only* when indicated in the descriptions.	`plotter_is` *window_name* `plotter_is PieChart`

* Some programs allow a wider range of special symbols in names than the period (.), hyphen (-), and underscore (_). However, you should avoid using the following symbols in names because they are reserved for other uses: slash (/), backslash (\), vertical bar (|), greater-than symbol (>), less-than symbol (<), semicolon (;), number sign (#), circumflex (^), ampersand (&), asterisk (*), apostrophe ('), double quotation mark ("), equals sign (=), question mark (?), opening parenthesis ((), closing parenthesis ()), opening bracket ([), and closing bracket (]).

† Wherever a *file_name* appears in this chapter, you may instead use a relative or absolute *path_name*.

The following programs from the Utilities disc enable you to make parameter selections interactively from the program window (once the program has started running), so that typing parameters in the PAM command line is unnecessary:

- **copy_disc**.
- **format_disc**.
- **rename_disc**.
- **verify_disc**.

Online Documentation

README Files. Each of the discs shipped with your computer includes at least one file named **README** that briefly describes the files on that disc. You can view or print a **README** file for an overview of the disc contents.

Documentation Folder. There are more than 30 programs on the HP-UX Commands disc. Documentation for these programs may be found on the Utilities disc in a folder named **hp-ux.doc**. To access this documentation, open the **hp-ux.doc** folder, then view or print the documentation files for the desired HP-UX program.

For example, for information on the Bourne shell program (*sh*), you can view or print the file **sh.doc** (complete path name is **/util/hp-ux/hp-ux.doc/sh.doc**).

Online Help. All the programs on the *Utilities disc* offer "help," or textual information, so that you can use the programs without having to refer to this chapter.

To use the online help of a program on the Utilities disc:

1. Insert the Utilities disc and ensure that PAM shows **/util** as the Open Folder.
2. Type the following in the PAM command line:

   program_name `-help`

   Note that you should type a space and a hyphen (-) before the word `help`.
3. Press (Return).

The program will create a window to display usage information, which may include parameter values and examples. You can use the instructions in this window to learn how to use the program.

In This Chapter

Following are summaries of Integral PC commands and programs, arranged alphabetically.

Autost

Name

Autost—Autostart this program

Syntax

Autost

Location

Tutor disc, Diagnostic disc

Description

The autostart capability of the computer enables PAM to start a program automatically when the computer is switched on.

After the computer has finished its self-test, PAM searches for a file named **Autost** in the outermost folder on the disc in the internal drive. If PAM finds the **Autost** file, then PAM will act on that file as if you had manually highlighted and started the file.

Note that the only difference between an **Autost** file and any other file is simply the name. (The name must be spelled with an uppercase **A** and lowercase **utost**.)

An **Autost** file will also begin executing:

- If PAM is stopped with the (Shift)(Stop) or (CTRL)(C) keystroke.
- If the computer is reset with the (Shift)(Reset) keystroke.
- If you start another version of PAM manually. The new version of PAM will search the current open folder for the **Autost** file.

...Autost

Autost files are typically PAM *script files* that cause the execution of other program files. Here are some sample scripts:

Sample Autost Script Files

Contents of the File	Description
`mt`	Causes the file named **mt** to begin executing in a window named **Autost**. The PAM window will not appear in the display until **mt** *finishes* executing.
`mt #Multiterrain`	Causes the file named **mt** to begin executing in a window named **Multiterrain**. The PAM window will not appear in the display until **mt** *finishes* executing.
`mt #Multiterrain &`	Causes the file named **mt** to begin executing in a window named **Multiterrain**. The PAM window will appear, although the **mt** window will be the active window.

An ampersand (&) in an *Autost* script file enables the PAM window to appear in the display. Without the ampersand, the PAM window will be suppressed until the *Autost* file runs to completion.

Example

```
Autost
```

Limitation

PAM will not autostart from any disc but the disc in the internal disc drive.

See Also

scan_discs

beep_off

Name

beep_off—disable the internal beeper

Syntax

beep_off
beep_off -help

Location

Utilities disc

Description

Beep_off disables the internal beeper so that programs will not be able to generate sounds from the computer.

A *beep_off* condition remains in effect until:

- You execute a *beep_on* command.
- You switch the computer off and back on.

Example

```
beep_off↵
```

See Also

beep_on

beep_on

Name

beep on—reenable the internal beeper

Syntax

beep_on
beep_on -help

Location

Utilities disc

Description

Beep_on causes the internal beeper to become accessible so that programs can cause the beeper to sound.

You need to use *beep_on* only after executing the *beep_off* program.

Example

```
beep_on
```

See Also

beep_off

Name

cd—change directory (change the open folder)

Syntax

cd
cd ..
cd *folder_name*

Location

Built into PAM

Description

The *cd* command causes a specific folder to be opened and its contents listed in the PAM window. The *folder_name* parameter identifies what folder to open. If the folder is not contained in the current open folder (that is, if you can't see the *folder_name* in the PAM list of `Folders`), then use a full path name to identify the folder.

With the exception of the first form of the command above (**cd** by itself), the two letters **cd** need not appear in a change directory command.

Examples. Entering `MyFolder` in the PAM command line is the same as entering `cd MyFolder`. Entering `..` is the same as entering `cd ..` to change directories (folders).

The **cd** command is included in PAM for the convenience of users familiar with the Bourne and csh shells.

Any folder currently included in the file system can be opened with the *cd* command and displayed by PAM unless its *read* or *execute* permission has been removed (see *chmod* in this chapter). If the folder specified cannot be located, PAM will display a message and keep the current folder open.

...cd

Entering *cd ..* (with two periods as the parameter) causes the current open folder to be closed and causes PAM to display the contents of the outer folder. Using *cd ..* is the equivalent of pressing Close (PAM key f8).

If *cd* by itself is entered in the PAM command line, then the system attempts to open the *home folder*. The home folder is part of the active environment, which is set at power on and set at the execution of a *getenv* PAM command. The absolute path name of the home folder is assigned to the variable HOME in the environment file. If the home folder is not defined or cannot be located, then the root folder of the file system is displayed.

Examples

```
cd CurrentFiles
cd /usr/CurrentFiles
cd ..
cd ../../CurrentFiles
```

See Also

chmod
getenv
Chapter 3 of this guide

Name

close—close the open folder

Syntax

close

Location

Built into PAM

Description

Entering the *close* command is the same as pressing Close (f8). The command causes PAM to close the current open folder and to display the files contained in the outer folder.

If you enter a *close* command when the open folder is /, the open folder will not change.

Example

```
close
```

See Also

cd
Chapter 3 of this guide
Chapter 6 of the tutor and user's guide

chmod

C

Name

chmod—change the mode (permissions) of a file or folder

Syntax

chmod *permission_code file_name ...*
chmod *permission_code folder_name ...*

Location

HP-UX Commands disc

Description

The *chmod* program changes any of the *permissions* associated with files or folders. These include *read, write,* and *execute* permissions.

You may include any number of *file_names* or *folder_names* in the command line. The same change will be made to each file or folder.

The *permission_code* determines what changes are made to the permissions of the files or folders. You can use *chmod* to:

- Remove *read* permissions from a file so that the file cannot be edited or viewed.
- Remove *write* permissions from a file to protect the file from accidental deletion or alteration.
- Restore *read* or *write* permissions to a file when you want to modify the file.
- Add *execute* permissions to a file so that the file can be used as a *script file.*

The *permission_code* consists of three connected symbols and takes the following form:

∃ *operator permission_name*

The a symbol stands for "all." The *operator* and *permission_name* symbols are defined as follows:

Chmod Operators

Symbol	Description
+	Add the specified permission to the file.
−	Remove the specified permission from the file.
=	Set the specified permission and remove all other permissions.

Chmod Permissions

Symbol	Description
r	Read permission. Enable users to view, list, print, and rename files. Enable users to open folders.
w	Write permission. Enable users to modify, copy, move, and delete files. Enable users to change the contents of folders.
x	Execute permission. Enable users to execute a file. Enable users to access the contents of folders.

Use the tables above when constructing the *permission_code* for the *chmod* program. For example, if you want to change the permission of a file so that it is read-only, then use the = operator. The command line will read:

```
chmod a=r file_name
```

As a second example, if you want to change an ordinary text file into a script file that PAM can execute as a series of commands, then use the + operator and the **x** permission name:

```
chmod a+x file_name
```

The permissions for folders work as follows:

- If you want to allow a user to execute or access a file within a folder but not to *open* the folder itself, then remove the folder *read* permision.
- If you want to secure all the files in a folder from deletion, name changes, or permissison changes, then remove the *write* permission from the folder.
- If you want to deny access to a folder altogether, then remove the *execute* permission from the folder.
- You can always rename a folder if the folder that contains the desired folder has *write* permission.

Notes on *chmod*:

- File permissions are checked with each action performed upon a file (such as renaming a file). Folder permissions are checked with each action performed upon a folder (such as deleting a file from a folder).
- Normally, data files are created with both read and write permissions, but not execute permission. Use **chmod a-r** or **chmod a-w** when you want to protect a data file from access by others.
- Normally, *chmod* does its work silently. It creates a window only when you've entered an incorrect form of the command.
- Use the **-ld** options with the *ls* program when you want to check the permissions of a folder.

Examples

```
chmod a+x print_it
chmod a+r+w Hello
chmod a-w /usr/NewFile /doc/Save.Me
```

Limitations

- *Chmod* doesn't signal PAM when changing a data file to an executable file or *vice versa*. You can use the PAM Reread key (f1) or the CTRL L keystroke to update the PAM window after executing *chmod*.
- Embedded blanks are not allowed in the *permission_code*. For example, **a+w** is a valid *permission_code*; **a + w** is not.
- The description of *chmod* on these pages is not complete. Please refer to the documentation listed below for more information.

See Also

ls
reread
/util/hp-ux.doc/chmod.doc on the Utilities disc

copy

C

Name

copy—copy a file

Syntax

copy *file_name*
copy *source_file_name destination_file_name*
copy *source_file_name destination_folder_name*
copy *source_file_name ... destination_folder_name*

Location

Built into PAM

Description

The *copy* command enables you to make extra copies of individual files.

If you specify only one parameter after the command name, then the action is the same as highlighting a file name in the PAM window and pressing Copy (f5).

That is, PAM will issue a `Please wait...` message while copying the file into the **/tmp** electronic disc folder. Afterwards, PAM will prompt you for the destination name with the message—`Now complete the copy with 'to file name'.`—and display the word `to` in the command line. After supplying the destination file name or folder name, press Return. PAM will copy the temporary file from **/tmp** to the destination and afterwards delete the temporary file from the electronic disc.

To cancel a copy operation at the point when PAM prompts you to `complete the copy`, erase the word `to` in the command line with the Back space key or Shift Delete line keystroke. (However, see Limitations below.)

Use the first form of the command when copying a file across discs in the internal disc drive. Insert the second disc in the internal disc drive when prompted to `complete the copy`, and specify the name for the duplicate file after the word `to`. (If you simply press Return, the name of the destination file on the second disc will have the same name as the source file.)

If you specify both source and destination, as in the second form of the command, then PAM will complete the copy operation without further prompting. Unless you specify a path name for the destination file, PAM places the duplicate file in the open folder.

Use the third form of the *copy* command when the destination folder for the file is different from the source folder. A copy of the file by the same name will be made in the destination folder.

The fourth form of the *copy* command provides an easy way to copy a number of files to another folder. List the names of the desired files after the *copy* command; each will be copied to the destination folder. Note that if any one of the copy operations fails, then PAM will display an error message and cancel the rest of the copy operation at that point. Finally, to copy a file from one disc to another when both discs are currently in the system, use path names to identify source and destination discs.

The message line will show `Done with 'copy'` when PAM finishes copying the file or files.

Normally, you shouldn't use the name of an existing file for the *destination_file_name*. If you do specify a file that already exists and if it has write permission (the default), then PAM overwrites the file with information from the source file, causing the loss of information originally in the file.

Note that the *copy_disc* program from the Utilities disc provides an easy way to copy an entire disc.

Examples

```
copy ThisFile
copy ThisFile ThatFile
copy ThisFile ../../ThatFolder
copy ThisFile AndThatOne AndTheOtherOne AnotherFolder
```

Limitations

- Entering **copy** by itself causes an error message.
- The *copy* command only allows you to copy individual files; you cannot use the command to copy a disc or a folder.
- You can cancel a copy operation by erasing the word `to` in the PAM command line; however, the copy will remain pending until you've started another copy, rename, or move operation. That is, if at some future point you enter `to` *file_name* in the PAM command line, then PAM will attempt to finish the previously started copy operation.
- If a file or folder permission has been changed with the *chmod* command, then a copy operation may not be successful. The following permissions are necessary for copying files:
  - **a.** A source file must have *read* permission.
  - **b.** If a source file is not in the open folder, then the folder for the file must have both *read* and *execute* permission.
  - **c.** The destination folder must have *write* permission.
  - **d.** If the destination folder is not the open folder, then that folder must have both *write* and *execute* permission.

See Also

chmod
copy_disc
Chapter 3 of this guide
Chapter 6 of the tutor and the user's guide

Name

copy_disc—copy the contents of one disc to another disc

Syntax

copy_disc
copy_disc *drive_name*
copy_disc *source_drive_name destination_drive_name*
copy_disc -help

Location

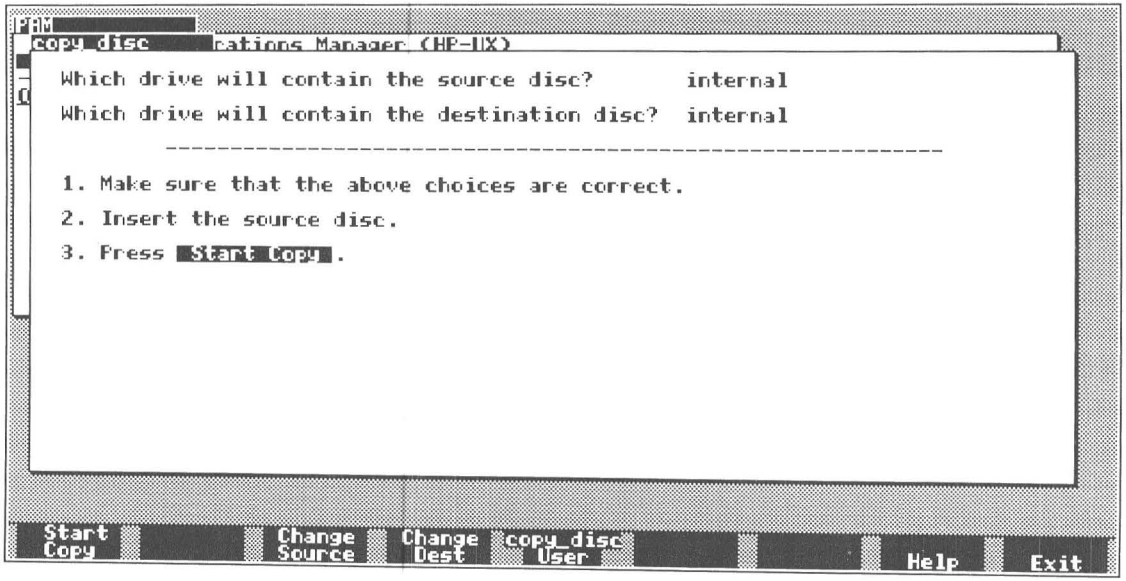
Utilities disc

Description

Copy_disc does a complete byte-by-byte copy of the data on one disc (the *source* disc) to another disc (the *destination* disc).

...copy_disc

If you start *copy_disc* without specifying any parameters, you'll create the following program window:



The *copy_disc* window shows that the internal disc drive will contain both the source and the destination discs.

The following function keys control the *copy_disc* program:

Function Keys for Copy_disc

Function key	Description
Start Copy (f1)	Causes the program to start the actual copying from the source disc to the destination disc. The copy operation can be canceled with Exit (f8).
Change Source (f3)	Enables you to enter a different *drive_name* for the source disc, such as `D000`. Pressing the key moves the cursor to the `source disc` field and clears that field. Type in the new source and press Return. To restore the field to its previous value, backspace over the field and press Return while the field is blank.
Change Dest (f4)	Enables you to enter a different *drive_name* for the destination disc, such as `D001`. Pressing the key moves the cursor to the `destination disc` field and clears that field. Type in the new destination and press Return. To restore the field to its original value, simply press Return while the field is blank.
Help (f7)	Creates a "help" window, separate from the **copy_disc** window, in which you can read usage information about the program.
Exit (f8)	Ends the program, cancels the current copy operation (if in progress), and eliminates the *copy_disc* program window.

If you specify just one *drive_name* in the command line, then that name will appear as both source and destination in the *copy_disc* window.

Notes on copy_disc.

- You must format a disc before you can copy information to it.
- *Copy_disc* will name the destination disc with the same name as the source disc.
- *Copy_disc* will prompt you to switch discs if the source and destination are the same. This means that the program will read as much of the source disc into memory as possible, and then ask you to insert the destination disc to complete the write. To reduce the number of disc "swaps," eliminate as many programs and windows as possible before running the *copy_disc* program.
- After completing a copy operation, *copy_disc* will display the total number of bytes copied. *The program will continue executing.* To copy another disc, follow steps 1 through 3 in the *copy_disc* window. To end the program, press Exit (f8).
- If you press Exit in the middle of a copy operation, the source disc will not be affected in any way, but the destination disc will be left in an undefined state and will probably be unusable.

Disc performance information. There are times when a file-by-file copy to another disc is preferable to a *copy_disc* operation. The reason is that after you have used a disc for a long time and have copied, moved, and deleted a number of files on that disc, the remaining files will be interspersed with "gaps" on the physical medium. The result is slower disc access times and decreased performance. The effect varies, depending on the number of copies, moves, and deletions the disc has undergone.

When you use *copy_disc*, the destination disc inherits the exact file organization of the source disc, including any gaps that exist on the source disc. If instead you use a combination of PAM *makefolder* and *copy* commands (perhaps in a PAM script file) to copy the files from one disc to another disc, the new disc will have its files recorded in an optimum amount of space.

To summarize, *copy_disc* provides an easy way to duplicate discs; a file-by-file disc copy involves more effort, but may result in enhanced performance for the destination disc.

Examples

```
copy_disc
copy_disc D000
copy_disc D000 D001
```

Limitations

- The discs must be the same size and type (for example, both single-sided or both double-sided).
- The (Back space) key is the only editing key that you can use to edit any of the fields in the *copy_disc* window.

See Also

format_disc
Chapter 5 of the tutor and the user's guide

Name

delete—delete a file or folder

Syntax

delete *file_name* **...**
delete *folder_name* **...**

Location

Built into PAM

Description

The *delete* command enables you to delete individual files and delete empty folders.

Once you delete a file, it is gone forever. There is no way to restore a deleted file. Be certain that you've finished using a file before deleting it.

Include the names of files and empty folders after the *delete* command and then press (Return). The PAM window will be updated to show that a deletion has occurred.

Note that Delete ((f7)) is a typing aid for the *delete* command. Pressing Delete causes the word *delete* to appear in the PAM command line, followed by the name that is highlighted in the PAM window.

To protect a file from accidental deletion, remove its *write* permission.

...delete

Examples

```
delete OldFile
delete OldFile OlderFile OldestFile
delete OldFolder
```

D

Limitations

- Entering `delete` by itself causes an error message.
- You must delete all the files in a folder before you can delete the folder itself. Also, the folder must have *read*, *write*, and *execute* permission before it can be deleted.
- A folder must have *write* and *execute* permission before any of the files in it can be deleted.
- You can't delete a file if its *write* permission has been removed.
- The *delete* command cannot be used to delete a disc, the open folder, the **/** folder, or the **rom** folder.
- If you include multiple file or folder names after a *delete* command and if any one of the deletions fails, then PAM will display an error message and cancel the delete operation at that point.

See Also

chmod
Chapter 3 of this guide
Chapter 6 of the tutor and the user's guide

Name

diagnostic—run the diagnostic test

Syntax

diagnostic

Location

Diagnostic disc

Description

The *diagnostic* program performs a complete check of computer hardware, including:

- Read-only memory (ROM), in which the HP-UX operating system resides.
- Random access memory (RAM), in which your programs run.
- The computer keyboard, including individual keys, connectors, and cable.
- The circuitry leading to the electroluminescent (EL) display.
- The internal printer.
- The internal speaker.
- The real time clock, for providing time and date functions.
- The internal disc drive.
- The internal Hewlett-Packard Interface Bus (HP-IB), for controlling peripherals.
- Optional plug-in interfaces, including the serial, current loop, GPIO, and HP-IL interfaces.

The *diagnostic* program clears the contents of computer memory. Do not run the test if you are in the middle of work that you haven't yet saved. Also, power down the computer after you are finished running the diagnostic program. Then power up to resume work.

...diagnostic

To run the *diagnostic* program:

1. Insert the Diagnostic disc in the internal disc drive.
2. Highlight and start the *diagnostic* program.
3. Follow the instructions provided by the program.

Example

```
diagnostic
```

Limitation

The *diagnostic* program clears computer memory so that it is not possible to run the program concurrently with other programs.

See Also

Chapter 1 of the tutor and user's guide

Name

disp_time_off—disable the display timeout period

Syntax

disp_time_off
disp_time_off -help

Location

Utilities disc

Description

Normally, the electroluminescent (EL) display turns off when the computer has been inactive for about 15 minutes to prolong the life of the display.

That is, the display will go blank after a timeout period if:

- No key has been pressed.
- No program output has occurred.

The *disp_time_off* program overrides the timeout period so that the display will remain turned on indefinitely.

A *disp_time_off* setting remains in effect until:

- You run the *disp_time_on* program.
- You switch the computer off and back on.

Example

```
disp_time_off
```

See Also

disp_time_on

disp_time_on

Name

disp_time_on

Syntax

disp_time_on
disp_time_on -help

Location

Utilities disc

Description

Use *disp_time_on* to restore the normal 15-minute timeout period for the electroluminescent (EL) display.

The *disp_time_on* program undoes the effect of the *disp_time_off* program.

Example

```
disp_time_on
```

See Also

disp_time_off

Name

echo—echo text to a window or a file

Syntax

echo
echo *text*
echo *text* **#** *window_name*
echo *text* > *file_name*
echo *text* >> *file_name*

Location

Built into PAM

Description

The *echo* command enables you to write a string of characters either to a window or to a file.

For example, entering `echo Hello, world! #MyWindow` in the command line causes PAM to create a window named `MyWindow` and to display the characters `Hello, world!` in that window. The window is an *alpha* window that you can modify using the typing and editing keys of the computer.

If the window specified after the **#** symbol already exists, then PAM writes the *text* beginning at the current cursor location in that window.

The redirection symbol > causes PAM to create the specified file (or to overwrite an existing file) and to enter the *text* beginning at column 1, line 1 of that file. For example, entering:

```
echo The rabbit has a habit >OneLiner
```

—causes PAM to create a file named `OneLiner` in the open folder and to write the characters `The rabbit has a habit` to that file. (Afterwards, press CTRL L to cause PAM to update the PAM window with the name of the new data file.)

...echo

The redirection symbol >> causes PAM to *append* the *text* to a file.

Don't confuse the *echo* command with the Echo key (f2). The command is used for writing strings to windows and to files; the key is used just as a typing aid for the PAM command line.

The main usefulness of *echo* is in PAM script files to display messages as the script files are executing. For example, after you've copied a file using the *copy* command in a script file, you can include a command line, such as `echo Done with copy!` to report on the disc copy completion.

If you enter an *echo* command without specifying a *window_name* or a *file_name*, then PAM will create an alpha window named `echo`.

Use single quotation marks (' ') or double quotation marks (" ") to enclose any *text* that includes special characters, such as a vertical bar (|). Quoting special characters means that PAM won't use them for anything but textual information. You may nest a pair of single quotation marks within a pair of double quotation marks in the same line, and *vice versa*.

Examples

```
echo 'How are you today, Sir?' #HereWeGo
echo "Done with the 'copy' command" >> logfile
```

Limitation

If *echo* directs output to a file, then that file must have *write* permission. If the file is not in the open folder, then the folder itself must have *write* and *execute* permission.

See Also

chmod
Chapter 3 of this guide

Name

fedit—font editor

Syntax

fedit
fedit *file_name*

Location

Standard Applications disc, in **/apps/fonts**

Description

The *fedit* program enables you to create and modify *fonts*, or character sets, for the computer. The idea is that you open an existing character set (or *font file*) for editing, and then use the cursor keys (or mouse) and the function keys to *change* individual characters and to *add* new characters to the font. Afterwards, you can permanently save the character set in a new font file and use that font file for the character set of any alpha window you choose.

You work on one character at a time by selecting individual dots in the *cell* for that character. You can easily view the results as you make changes.

For example, if you wanted to change the style of the character A, then you would:

1. Select the character A from the character set matrix in the *fedit* window.
2. Point at and turn on/off individual dots in the expanded view of the character A.
3. Examine the new character A by typing (Shift)(A) from the keyboard.
4. When satisfied with your changes, save the changes in a new file with the ? Files and the Store keys.
5. Load the new font into a window with the *set_font* program.

If you enter the *fedit* command by itself, then a copy of the system font (or character set) will be loaded for editing. (Note that you can't *change* the system font). If you enter the *fedit* command followed by a *file_name*, then that font file will be loaded for editing.

...fedit

Example

```
fedit block12x16
```

Limitations

- Using the *fedit* program is easy, but creating a completely new character set can be a lengthy process, because each of the 256 characters must be individually edited.
- The maximum size for a new font is 16 dots wide by 16 dots high. (The size of the display itself is 512 dots wide by 256 dots high.)

See Also

set_font
/apps/fonts/fedit.doc on the Standard Applications disc

Name

format_disc—format a disc for use

Syntax

format_disc
format_disc *disc_name*
format_disc *disc_name drive_name*
format_disc -help

Location

Utilities disc

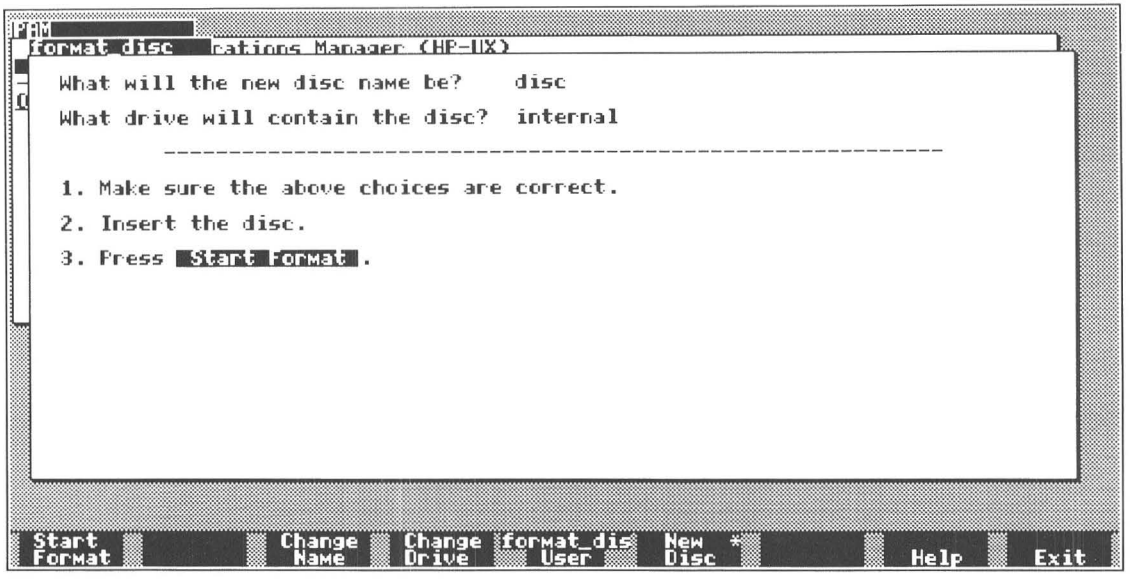
Description

Format_disc prepares new discs for storing information.

The *format_disc* program destroys any information that already exists on a disc. Do not use *format_disc* unless you're certain that nothing of value resides on the disc that you intend to format.

...format_disc

If you start *format_disc* without specifying any parameters, you'll create the following program window:



The *format_disc* window shows that the program will format the disc in the internal disc drive and will name that disc **disc**.

The following function keys control the *format_disc* program:

Function Keys for Format_disc

Function key	Description
Start Format (f1)	Causes the actual formatting to begin. Make sure that the `drive name` field is correct before pressing this key!
Change Name (f3)	Enables you to specify a different *disc_name* for the disc that you will format. (The default name is simply `disc`.) Pressing the key moves the cursor to the `disc name` field and clears that field. Type in the desired name (up to six characters) and press Return. To restore the field to its previous value, backspace over the field and press Return when the field is blank.
Change Drive (f4)	Enables you to specify a different *drive_name* for the disc drive that will do the formatting. Pressing the key moves the cursor to the `disc drive` field and clears that field. Type in the name of the disc drive, such as `D000`, and press Return. To restore the field to its previous value, backspace over the field and press Return when the field is blank.
New Disc (f5)	Enables you to choose between: ■ Initializing and then formatting a disc (the default), or ■ Formatting a disc only (a time saver).
Help (f7)	Creates a "help" window, separate from the *format_disc* window, in which you can read usage information about the program.
Exit (f8)	Ends the program, cancels the current format operation (if in progress), and eliminates the *format_disc* program window.

Remember to insert the correct disc before starting the formatting operation. Otherwise, you'll format the wrong disc!

...format_disc

Notes on *format_disc*.

- If you cancel a disc formatting operation, the disc will be left in an undefined state. The disc will need to be reformatted before it can be used again.
- If you specify the *disc_name* or the *drive_name* when entering the command, then the fields in the *format_disc* window will be set to those values. Note that you must include *both* parameters to specify an external disc drive, for example:

```
format_disc RES-2 D020
```

- After completing a format operation, *format_disc* will display the total number of bytes available on the disc medium. *The program will continue executing.* To format another disc, follow steps 1 through 3 in the *format_disc* window. To end the program, press Exit (f8).

E
F
G

Format_disc operates in two phases:

- The first phase *initializes* the disc medium. This phase may take a long time, especially when you're initializing a hard disc (up to a half-hour!) Track and sector marks are written on the disc media. At the same time, the disc is tested for bad sectors and tracks; if necessary, track "sparing" is performed, to prevent these bad areas on the disc from being used.
- The second phase *formats* the disc medium. This phase is much shorter. A *file system* with the specified disc named is created on the disc.

You don't always have to initialize a disc medium before formatting it. Once you initialize a new disc, you never need to initialize it again. Use New Disc (f5) to skip the lengthy initializing phase if you simply want to erase the contents of a used disc for a fresh start.

Initially, the New Disc * key label appears with an asterisk (*), indicating that *format_disc* is ready both to initialize and format a disc. Pressed once, the New Disc function key will cause *format_disc* to format the disc *only*—the asterisk (*) in the key label will turn off. Pressed twice, the New Disc * key will cause *format_disc* both to initialize and format, indicated by the reappearing asterisk.

Format_disc automatically calculates the size of a disc medium (measured in 1,024-byte blocks) and the number of files that the disc can hold. The numbers will be reported at the end of the formatting phase.

Examples

```
format_disc
format_disc SALES
format_disc SALES /dev/D030
```

Limitations

- The (Back space) key is the only editing key that you can use to edit any of the fields in the *format_disc* window.
- The initializing phase of a *format_disc* operation will slow down system performance. Try not to use the computer for running other programs when initializing discs.

Do not eject a disc during a disc formatting operation (while the red busy light is lit). Such action will not only interrupt the operation, but will cause the entire computer to stop functioning until you switch the power off and back on. To cancel a disc formatting operation, press Exit ((f8)).

getenv

Name

getenv—get environment (set the current PAM environment)

Syntax

getenv *file_name*

Location

Built into PAM

E
F
G

Description

The `getenv` command enables you to change the *environment* of PAM and of programs started from the PAM window. The *file_name* parameter identifies an *environment file*—a special text file that consists of environment *variables* and *settings*. The PAM environment is set with the variables in this file. The `PATH` variable, for example, defines the search path as a list of folders.

PAM Environment Variables

Variable Name	Description
HOME	Sets the folder that will be opened when a **cd** command is executed without parameters.
PATH	Sets the folders that PAM will search when you enter the simple name of a program in the PAM command line.
TERM	Identifies the terminal type to be used for alpha windows.
TERMCAP	Identifies terminal control sequences that applications can use to control alpha windows.
TZ	Identifies the time zone for the computer. The format is *xxxnzzz*, where *xxx* is a three-character abbreviation for the local time zone, *n* is the difference in hours from GMT, and *zzz* is a three-character abbreviation for the local daylight-savings time zone, if any. *N* may range from `12` through `-13` (use positive values for areas *west* of Greenwich and negative values for areas *east* of Greenwich). In Oregon, for example, the complete time zone string reads: `TZ=PST8PDT`.

You don't *need* to use *getenv* to set the PAM environment: At power on, PAM searches the disc in the internal disc drive for a file named **.environ** located in the outermost folder on the disc. If PAM finds **.environ**, then PAM uses that file to set the environment. Otherwise, PAM uses a file named **/rom/.environ**—located in read-only memory—as the default environment file.

The **/rom/.environ** file consists of the following settings:

The most common reason for changing the default PAM environment is to reassign the PATH variable—that is, to set the list of folders that PAM will use to find programs as well as the *order* that PAM will search them.

Another reason to change the PAM environment is to change the TZ (time zone) variable, whose default is EST/EDT (Eastern Standard Time/Eastern Daylight Time). For example, the *date* and *ls* program on the HP-UX Commands disc rely on the value of the TZ variable.

To create a new environment file, you can copy **/rom/.environ** to a new disc file (say, **newenvironment**) and then use a text editor to modify the new environment file. Note that the text editor (such as *vi*) must enable you to enter lines *longer that 80 characters*, since both the PATH and the TERMCAP settings require long lines.

It shouldn't ever be necessary to change the TERM or the TERMCAP variable.

Example

```
getenv myfile
```

Limitations

- You cannot set individual environment variables directly from the PAM command line. You must instead use a text editor to create and modify a complete environment file and afterwards set *all* environment variables with the *getenv* command.
- The *getenv* command will have no effect on PAM if the command is included in a script file. The reason is that script files run in their own environment, separate from the PAM environment.
- An environment file must have *read* permission. The folder that contains the environment file must have *execute* permission.

See Also

cd
chmod
Chapter 3 of this guide

Name

load_hpgl—load the HP-GL command set for plotter windows

Syntax

load_hpgl

Location

System disc

Description

The *load_hpgl* program enables the computer to use the Hewlett-Packard Graphics Language (HP-GL) to plot data in graphics windows. When you run *load_hpgl*, you cause the HP-GL *command set* (from the file **hpgl.data** on the System disc) to be loaded into system memory.

Afterwards, any plotter window that the computer creates (as with the *plotter_is* command) will "understand" HP-GL plotter commands and be able to plot data accordingly.

Notes on load_hpgl:

- You should run the *load_hpgl* program *before* creating a graphics window in order to use HP-GL commands within the window.
- You do *not* need to use *load_hpgl* to control external Hewlett-Packard plotters, because the HP-GL command set has been built into the plotters.
- Some programs, such as BASIC, will load the HP-GL command set for you, so that running *load_hpgl* is unnecessary.
- Once loaded, the HP-GL command set will remain in system memory until the computer is switched off.

An error will be reported if the file containing the HP-GL command set (named **hpgl.data**) cannot be found.

...load_hpgl

Example

```
load_hpgl
```

Limitations

- The *load_hpgl* program will fail if there is insufficient memory to load the HP-GL command interpreter.
- If you run *load_hpgl* more than once before switching off the computer, then multiple copies of the HP-GL command interpreter will be loaded into system memory. The result is wasted (unused) system memory.

See Also

plotter_is
Documentation for HP-GL commands

Name

load_serial—load the control program for the HP 82919A Serial Interface

Syntax

load_serial

Location

System disc

Description

The *load_serial* program enables the computer to control the HP 82919A Serial Interface. When you run *load_serial*, you cause the control program for the serial interface to be loaded into system memory. (This control program is often referred to as a *device driver*.)

The correct order of events is:

1. Physically install the serial interface in the computer.
2. Switch on the computer.
3. Insert the System disc in the internal disc drive.
4. Highlight and start the *load_serial* program.

Afterwards, to start using a serial printer or serial plotter:

1. Connect the printer or plotter device to the serial interface using the correct cable and connectors, and switch on power to the peripheral.
2. Insert the Utilities disc in the internal disc drive.
3. Run the *printer_is* or *plotter_is* program after supplying the appropriate parameters in the PAM command line.

Once loaded, the control program for the serial interface will remain in the system until the computer is switched off.

...load_serial

Note that an application program such as Datacomm can load the serial interface control program for you, so that running *load_serial* is unnecessary.

Regardless of the number of serial interfaces connected to the computer (up to 10 are possible), only one version of the control program needs to reside in system memory. *Load_serial* will create a place for each serial interface in the **/dev** folder, and give each interface a device name beginning with **cul**. The first serial interface will be named **cul00**; the second will be named **cul01**; and so on up to **cul09**.

An error will be reported if the file contining the control program for the serial interface (named **serial.data**) cannot be found.

Example

```
load_serial
```

Limitations

- If you run *load_serial* without having installed a serial interface, the program will display an error message, but load the control program anyway.
- If you run *load_serial* more than once before switching off the computer, then multiple copies of the control program will be loaded into system memory and will interfere with each other and perhaps interfere with the proper operation of the computer.

See Also

printer_is
plotter_is
Documentation for the HP 82919A Serial Interface

Name

ls—list the contents of folders

Syntax

ls *options file_name* **...**
ls *options folder_name* **...**

Location

HP-UX Commands disc

Description

Use *ls* to list more information about a file or a folder than is shown in the PAM window.

For each *file_name*, *ls* lists the name and other information specified by the *options*. For each *folder_name*, *ls* lists the contents of that folder, according to the *options*. By default, *ls* sorts the output alphabetically. When no names are supplied, *ls* lists the contents of the open folder.

The more common *options* are:

Ls Options

Symbol	Description
-l	List files using a long format, showing the permissions (read, write, and execute), size (in bytes), and time of last modification.
-t	Sort by time of last modification (latest first), instead of by name.
-a	List all entries, including "hidden" files (those whose names begin with a period).
-F	Cause directories to be marked with a trailing slash (**/**) and executable files to be marked with a trailng asterisk (*****).
-R	List the contents of folders contained within other folders (list recursively).
-d	List information (permissions, times, etc.) about folders.

The important information to be gained from running *ls* are the *sizes* of files (in bytes), the modification *times*, and the file *permissions*. File and folder permissions are listed as follows when you specify the **-l** option.

File and Folder Permissions

Symbol	Description
r	The file is readable.
w	The file is writable.
x	The file is executable. (For a folder, "execute" permission means permission to search the folder for a specified file.)
-	The indicated permission is *not* granted.

After you apply *chmod* to a file or a folder, you can observe the effects with the **ls -l** option. The file permissions will be set or cleared according to the options specified in *chmod*.

When you apply the **-l** option by itself to a folder, the *ls* program will report the total size of everything *in* the folder (measured in 1K-byte blocks), before listing the contents of the folder.

Examples

```
ls -al
ls -l File1 Folder1
ls -lR
ls -t | print
ls -F /Disc1
ls -ld ThisFolder ThatFolder
```

Limitations

- The output device is assumed to be 80 columns wide.
- The description of *ls* on these pages is not complete. Please refer to the documentation listed below for more information.

See Also

chmod
/util/hp-ux.doc/ls.doc on the Utilities disc

Name

makefolder—make a new folder

Syntax

makefolder *folder_name* **...**

Location

Built into PAM

Description

The *makefolder* command enables you to create new folders for storing files and other folders.

Each *folder_name* you enter as a parameter results in a new folder, created by default in the open folder. To create a folder in a different place, specify the path name for the new folder, such as `/Disc1/FolderHere`.

New folders will be created with *read*, *write*, and *execute* permissions. Use the *chmod* program to change folder permissions if you want to secure your information.

PAM will respond, `Cannot create '`*folder_name*`'.`, if you've specified the name of a file or or folder that already exists in the same place. (However, you may have duplicate names in separate folders.)

...makefolder

Examples

```
makefolder New
makefolder Here /There ../Everywhere
```

Limitations

- If you include multiple folder names after a *makefolder* command and if any one of the folders can't be created—for one reason or another—then PAM will display an error message and cancel the rest of the operation at that point.
- The folder that is to contain a new folder must have *execute* and *write* permissions.

See Also

chmchmod
Chapter 3 of this guide
Chapter 6 of the tutor and user's guide

Name

mount_disc—mount a flexible disc in an external disc drive

Syntax

mount_disc *drive_name*
mount_disc -help

Location

Utilities disc

Description

The *mount_disc* program is used for one reason—to add a flexible disc to the computer file system so that the computer can access it properly. The process is called *mounting a disc.*

You do *not* need to use *mount_disc*:

- If the disc is already inserted in the disc drive when the computer is powered up.
- When you change a disc in the *internal* disc drive.

You *do* need to use *mount_disc* any time that you want to insert a flexible disc in an external disc drive after switching on the computer.

...mount_disc

To mount a flexible disc in an external disc drive:

1. Run the *unmount_disc* program for the external disc drive. For example, to unmount the disc in **/dev/D000**, you could insert the utilities disc in the internal disc drive and enter `/util/unmount_disc D000` in the PAM command line. (**Note:** Skip this step and step 2 if you've powered up the computer before inserting a disc in the external disc drive.)
2. Remove the flexible disc from the external disc drive.
3. Insert the replacement disc.
4. Execute the *mount_disc* program, specifying the same disc drive.

The disc will be mounted at the root of your file system, so that the disc name will appear in the **/** folder.

Example

```
mount_disc D000
```

Limitation

You can mount only one disc at a time.

See Also

scan_discs
unmount_disc
Chapter 5 of this guide

Name

move—move a file or a folder to another folder

Syntax

move *file_name*
move *folder_name*
move *source_file_name destination_file_name*
move *source_file_name* **...** *destination_folder_name*
move *source_folder_name destination_folder_name*

Location

Built into PAM

Description

The *move* command enables you to:

- Move individual files from one folder to another.
- Move an individual folder and all of its contents to another folder on the same disc.

You can think of the *move* command as a combination of two other PAM commands—*copy* and *delete*. That is, a move operation is equivalent to first copying the source file to a destination and then deleting the source file.

If you specify both source and destination names after a *move* command, then PAM will complete the move operation immediately.

If you specify only one parameter after the command, then the action is the same as highlighting a name in the PAM window and pressing Move (f4).

That is, PAM will prompt you for the destination name with the message—`Now complete the move with 'to file name',`—and display the word `to` in the command line. After supplying the destination, press Return.

If you specify a *folder_name* as the destination, then the source file will be put in that folder under the same name as the original file. If you specify a *file_name* as the destination, as in the third form of the command, then the source file will overwrite an existing file by that name.

You can include the names of as many files as you'd like after the *move* command. All will be moved to the destination folder (which should appear as the last name in the command line).

The message line will show `Done with 'move'` when PAM finishes moving the files.

To cancel a move operation at the point when PAM prompts you to `complete the move`, erase the word `to` in the command line with the Back space or Shift Delete line keys. (However, see Limitations below.)

Moving a *folder* means that the folder and all the files and folders contained within that folder are moved to the destination folder. The organization of the files and folders *within* the source folder remains intact. You are not allowed to move a folder to another disc, including the electronic disc (**/** and **/tmp**), but you can move a folder to any other folder on the same disc.

Note that if the destination folder doesn't exist, then the last form of the command is equivalent to renaming the source folder.

Examples

```
move ThisFile ThatFolder
move ThisFile ThatFile
move ThisFile /Disc1
move ThisFile /usr/bin/NewFile
move File1 File2 File3  NewFolder
move ThisFolder ThatFolder
move ThisFolder ..
```

Limitations

- Entering *move* by itself causes an error message.
- You can cancel a move operation by erasing the word `to` in the PAM command line; however, the move will remain pending until you've started another move, copy, or rename operation. That is, if at some future point you enter `to` *file_name* in the PAM command line, then PAM will complete the previously started move operation.
- The *move* command cannot be used to move a folder across discs.
- The *move* command will not allow you to move the open folder.
- If you include multiple file names after a *move* command and if any one of the move operations fails, then PAM will display an error message and cancel the rest of the *move* command at that point.

See Also

copy
Chapter 3 of this guide
Chapter 6 of the tutor and user's guide

Name

plotter_is—set the current plotter device

Syntax

plotter_is *window_name*
plotter_is *HP-IB_address*
plotter_is *RS-232_device_name optional_parameters*
plotter_is -help

Location

Utilities disc

Description

The *plotter_is* program defines the standard plotter device (named **/dev/plotter**) that subsequent programs will use to plot graphics data. That is, once you've run *plotter_is* and afterwards started a graphics program, that program will direct any **/dev/plotter** output to the specified window or external plotter.

The plotter definition will remain in effect for the duration of the graphics program. Running *plotter_is* again will affect only programs started afterwards.

The *window_name* parameter specifies a new or existing *graphics* window (as opposed to an alpha window).

Specifying a new *window_name* causes the *plotter_is* program to create but not to *show* a graphics window by that name. You can both *show* and *select* this graphics window by pressing (CTRL)(Select). Then, to access the graphics menu for that window, switch to the System menu and press Graph (System menu key (f5)). The window will also show itself when a program first directs its output to the window.

Note that most graphics applications create their own windows so that you don't have to. The *plotter_is* program is most useful to direct program output to an external plotter.

...plotter_is

You may include one or more of the following parameters (in any order) when specifying *optional parameters* for a serial plotter.

Parameter	Values
baud rate	110, 300, 600, 1200, 2400, 4800, or 9600
parity	**odd**, **even**, or **no-parity**
pacing	**xon-off**, **hardware-handshake**, or **no-pacing**

If you omit the *optional_parameters*, the plotter will be set to **9600** baud, **no-parity**, and **no-pacing**.

Examples

```
plotter_is SalesFigures
plotter_is 5
plotter_is cul01 1200 even xon-off
```

See Also

load_serial

Name

print—print a file

Syntax

print *file_name ...*

Location

Built into PAM

Description

The `print` command causes files to be printed on the current printer device (defined as **/dev/lp**). Enter the names of the files to be printed as parameters.

If the printer is currently busy or multiple files are specified for printing, then the files to be printed are put in a list of waiting files. When the printer finishes printing the current file, it moves to the next top of form and begins printing another file in the list of waiting files. To cancel a print operation that's already started, use the PAM *stopprint* command.

Printing of the file is canceled if the system cannot access the disc the file is stored on. You can copy files to be printed to the root folder manually before executing the `print` command if all files to be printed cannot be included in the file system at the same time.

You can also "pipe" the output of one program into the *print* command using the vertical bar (`|`) symbol in the PAM command line. For example, entering `/hp-ux/ls -al | print` causes the *ls* program on the **/hp-ux** disc to direct its output to the current printer device via the *print* command.

Examples

```
print MyFile
print /util/hp-ux.doc/vi.quick.doc
print File1 File2 File3
ls -al | print
```

Limitations

- Entering **print** by itself causes an error message.
- If the disc containing the file currently being printed is removed from the disc drive, then printing of that file only continues until that portion of the file already in system memory (called the *print buffer*) is completed.
- If an out-of-paper condition occurs during a *print* operation, then the print buffer of the internal printer will fill up with 1K of characters. The computer will suspend the print operation until you insert paper in the printer and press the light-grey printer button. If another program besides PAM experiences an out-of-paper condition while printing to the internal printer, then—after the print buffer fills up—program execution will be suspended altogether. (Other program windows will not be affected unless they also attempt print operations during the out-of-paper condition.)

See Also

printer_is
stopprint
Chapter 3 of this guide

Name

printer_is—set the current printer device

Syntax

printer_is internal
printer_is *HP-IB_address*
printer_is *RS-232_device_name optional_parameters*
printer_is -help

Location

Utilities disc

Description

The *printer_is* program enables you to access external HP-IB and serial printers.

The *printer_is* program defines the standard printer device (named **/dev/lp**) that subsequent programs will use to print text. That is, once you've run *printer_is* and afterwards started another program, that program will direct any **/dev/lp** output to the specified printer.

The printer definition will remain in effect for the duration of the program. Running *printer_is* again will affect only programs started afterwards as well as subsequent PAM *print* commands.

The result is that programs started after the *printer_is* program will normally use the specified printer device for their print operations.

Specifying **internal** causes the *printer_is* program to restore the internal printer as the current printer.

...printer_is

You may include one or more of the following parameters (in any order) when specifying *optional parameters* for a serial printer.

Parameter	Values
baud rate	**110**, **300**, **600**, **1200**, **2400**, **4800**, or **9600**
parity	**odd**, **even**, or **no-parity**
pacing	**xon-off**, **hardware-handshake**, or **no-pacing**

If you omit the *optional_parameters*, the printer will be set to **9600** baud, **no-parity**, and **no-pacing**.

Examples

```
printer_is internal
printer_is 0
printer_is cul00 1200 even xon-xoff
```

Limitation

The *printer_is* program will not allow you to direct printer output to a window or file. (If you enter a destination parameter that is not a recognized printer device, the program will define the *internal* printer as the current printer device.)

See Also

load_serial
print

Name

rename—rename a file or folder

Syntax

rename *file_name*
rename *folder_name*
rename *source_file_name destination_file_name*
rename *source_folder_name destination_folder_name*

Location

Built into PAM

Description

The *rename* command enables you to change the name of an individual file or a folder.

If you specify only one parameter after the command, the action is the same as highlighting a name in the PAM window and pressing Rename (f6).

That is, PAM will prompt you for the destination name with the message—
`How complete the rename with 'to file name'.`—and display the word `to` in the command line. After supplying the destination file name or folder name, press Return.

If you specify both the source and destination names, then PAM will complete the renaming operation without further prompting.

The message line will show `Done with 'rename'` when PAM finishes renaming the file or folder.

Note that PAM will assume that the renaming is to take place within the *open folder*. If you wish to rename a file or folder contained in another folder, then you must specify a path name for *both* source and destination names.

...rename

To cancel a rename operation at the point when PAM prompts you to `complete the rename`, erase the word `to` in the command line with the Back space or Shift Delete line keys. (However, see Limitations below.)

Note that you need to use the *rename_disc* program from the Utilities disc to rename a disc.

Examples

```
rename OldFile
rename OldFile NewFile
rename ../OldFile ../NewFile
```

Limitations

- Entering **rename** by itself causes an error message.
- You can cancel a rename operation by erasing the word `to` in the PAM command line; however, the renaming will remain pending until you've started another rename, copy, or move operation. That is, if at some future point you enter `to` *file_name* in the PAM command line, then PAM will complete the previously started rename operation.
- The *rename* command cannot be used to rename a disc, the open folder, the **/** folder, or the **/rom** folder.
- If you include multiple file or folder names after a *rename* command and if any one of the rename operations fails, then PAM will display an error message and cancel the rest of the *rename* command at that point.

See Also

P
R

rename_disc
Chapter 3 of this guide

Name

rename_disc—change the name of a disc

Syntax

rename_disc
rename_disc *disc_name*
rename_disc *disc_name drive_name*
rename_disc -help

Location

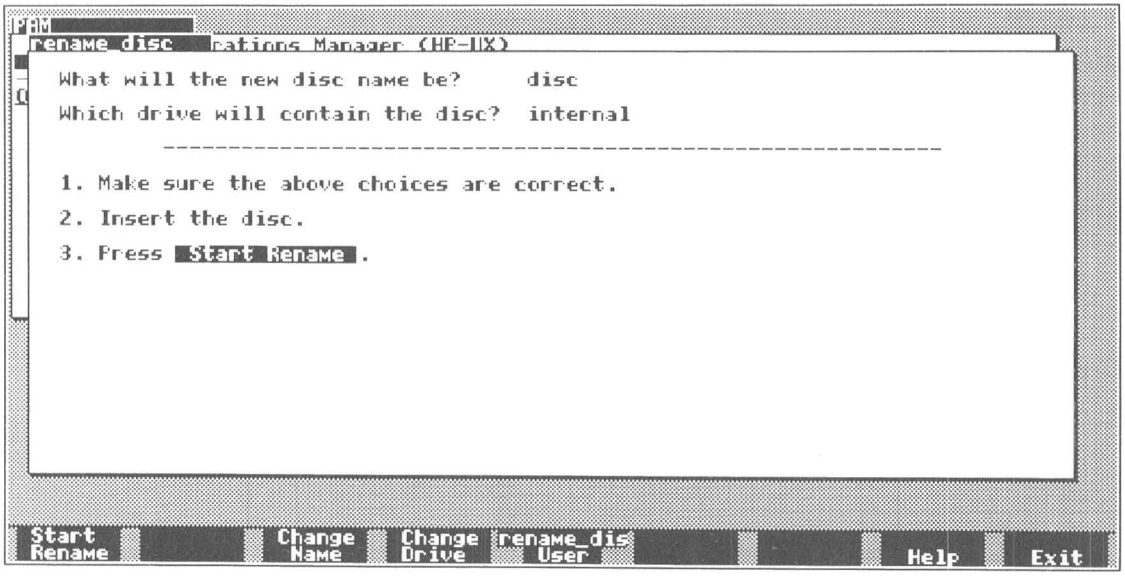
Utilities disc

Description

Rename_disc enables you to change the name of a flexible disc or hard disc.

...rename_disc

If you start *rename_disc* without specifying any parameters, you'll create the following program window:



The *rename_disc*

The *rename_disc* window shows that the program is ready to change the name of the disc in the internal drive to **disc**.

The following function keys control the *rename_disc* program:

Function Keys for Rename_disc

Function key	Description
Start Rename (f1)	Causes the actual renaming to occur.
Change Name (f3)	Enables you to specify a different *disc_name* for the disc. (The default name is simply `disc`.) Pressing the key moves the cursor to the `new disc name` field and clears that field. Type in the desired name (up to six characters) and press Return. To restore the field to its previous value, backspace over the field and press Return when the field is blank.
Change Drive (f4)	Enables you to specify a different *drive_name* for the disc drive that will do the renaming. Pressing the key moves the cursor to the `disc drive` field and clears that field. Type in the name of the disc drive, such as `D000`, and press Return. To restore the field to its previous value, backspace over the field and press Return when the field is blank.
Help (f7)	Creates a "help" window, separate from the *rename_disc* window, in which you can read usage information about the program.
Exit (f8)	Ends the program, cancels the current rename operation (if in progress), and eliminates the *rename_disc* program window.

...rename_disc

Notes on *rename_disc*.

- If you specify the *disc_name* or the *drive_name* when entering the command, then the fields in the *rename_disc* window will be set to those values. (Note that you must include *both* parameters to specify an external disc drive.)
- After completing a rename operation, *the rename_disc program will continue executing*. To rename another disc, follow steps 1 through 3 in the *rename_disc* window. To end the program, press Exit (f8).

Examples

```
rename_disc
rename_disc Wilhel
rename_disc Wilhel D000
```

Limitation

The Back space key is the only editing key that you can use to edit any of the fields in the *rename_disc* window.

Name

reread—reread the open folder

Syntax

reread

Location

Built into PAM

Description

The *reread* command causes PAM to check the contents of the current open folder and to update the PAM window with that information.

Entering *reread* in the PAM command line is the same as highlighting the name of the open folder and pressing **Reread** (f1).

Note also that pressing CTRL L is the same as entering the *reread* command.

Use *reread* when you've made changes to one or more files in the open folder and you want to ensure that the list of files and folders in the PAM window is properly updated.

For example, if you've changed a text file to a PAM script file using the *chmod* command, you need to cause PAM to update the window in order to verify that the *data* file has become a *program* file.

Example

```
reread
```

See Also

chmod
Chapter 3 of this guide

rpn

Name

rpn—Reverse Polish Notation calculator

Syntax

rpn

Location

Standard Applications disc

Description

The *rpn* program serves as a personal use calculator. The *rpn* window is small enough so that you can keep the calculator around in the display while running other applications.

The calculator uses the same form of logic as other Hewlett-Packard calculators—Reverse Polish Notation. To calculate results, you enter first one operand, press (Return), enter the other operand, and then press the operator key (such as (−) for subtraction).

Functions available from the keyboard include addition ((+)), subtraction ((−)), multiplication ((*)), division ((/)), and percent ((%)). Use the numeric keypad at the righthand side of the keyboard for easy digit entry.

For example, to calculate $\mathbf{13.6 \times 10^{-6}}$:

1. Type **23.8**.
2. Press (Return) to enter the first operand.
3. Type **13.6**.
4. Press (E) for *exponent*.
5. Type **6**, then press chs ((f1)).
6. Press (Shift)(*) to multiply.

The answer will be displayed in the *X*-register of the calculator. If you're familiar with Hewlett-Packard calculators, you'll recognize the visual display of the *X*, *Y*, *Z*, *T*, and *L* (last *X*) registers in the calculator window.

The function key labels for the calculator indicate other calculator functions, such as **chs** (f1), to change the sign of the number in the *X*-register.

To change the number of digits shown, use the f key and a digit key. For example, to display four decimal places, press f followed by 4. (The "f" stands for "fixed point.")

The **send** key operates similarly to the PAM **Send** key. Its purpose is to send results from the X-register to other program windows, for example, to send a calculation to a spreadsheet window.

Example

```
rpn
```

Limitations

- The calculator provides only one storage register, accessed with function keys f5 and f6.
- The calculator is not programmable.

See Also

send
The owner's manual for any Hewlett-Packard calculator

Name

scan_discs—scan for new disc drives

Syntax

scan_discs

Location

/rom/scan_discs

Description

The *scan_discs* program enables you to add new disc drives to the computer after you've already powered up the computer.

Normally, *before* switching on the computer, you should switch on all external disc drives (in any order), and allow them to complete their warmups and self-tests.

Then when you switch on the computer, the *scan_discs* program is run automatically to find the disc drives connected to the computer. If you're using one or more external disc drives, you may notice:

- The busy lights of the disc drives will turn on and off, perhaps several times.
- The PAM window will be updated one or more times.

The reason for this activity is that *scan_discs* is locating the drives and creating names for the drives in the **/dev** folder so that you can access the drives later on.

Please be patient while the computer is powering up. If you start using PAM before the *scan_discs* program has completed its check of available disc drives, then PAM may not be able to access a specified disc. Within a few moments, when *scan_disc* finishes executing, PAM *will* be able to find all of your discs.

...scan_discs

If there is a disc in the internal disc drive during the power up procedure, then:

1. PAM will read and process the lines of the **.environ** file, if found on the disc.
2. PAM will load and start executing the **Autost** file, if found on the disc.
3. PAM will open the outermost folder on the disc.

If there is no disc in the internal disc drive but there are discs in the external drives, then *scan_discs* will cause PAM to open the outermost folder on the disc in the drive that has the *lowest-numbered* HP-IB address.

After the computer has completed its power up procedure, you can examine the **/dev** folder to check the *drive_names* of the disc drives that *scan_discs* has located (such as **D000** and **D001**). The drive names will correspond to physical devices according to the addressing scheme described in Chapter 5 of this guide.

If you want to add a disc drive to the system after you've powered the computer, then you can run *scan_discs* "manually." Enter the simple name (**scan_discs**) or the complete name (**/rom/scan_discs**) of the program. Although the program does its work silently, you may note its progress by observing the disc drive lights.

Afterwards, you can check the new drive name by examining the **/dev** folder and check the *disc name* of the disc in the drive by examining the **/** folder.

Example

```
scan_discs
```

Limitations

- If you have a number of external disc drives connected to the computer, then *scan_discs* may take a long time to finish executing.
- *Scan_discs* is useful only for adding external disc *drives* (the hardware) to the system. Its purpose is *not* to add individual *discs* to the system (the software). If you want to do the latter after you've already powered the computer, you should use the *mount_disc* program from the Utilities disc.

See Also

Autost
getenv
mount_disc
unmount_disc
Chapter 5 of this guide

send

Name

send—send a path name to a window

Syntax

send

Location

Built into PAM

Description

The *send* command enables you to send the complete path name of a file from the PAM window to another window.

What happens is that the *highlighted file name* is appended to the *open folder name* and sent as a string of characters to the window immediately behind the PAM window. The complete path name is followed by a carriage return. The result is the same as if you had typed the complete path name in the destination window yourself and pressed (Return).

The *send* command is helpful when you're running an application that requests a file name.

For example, when MemoMaker prompts you for a file name for editing, you can quickly select the PAM window, highlight the name of the file that you want to edit, and *send* the name to the MemoMaker window. Without further prompting, MemoMaker will open the specified file.

Entering the *send* command in the PAM command line is the same as pressing Send (f3). The handheld mouse works well in manipulating windows and sending path names to particular applications with the Send key label.

If there are no windows besides PAM showing in the display, then the *send* command is ignored.

Example

```
send
```

Limitation

If you have a number of windows in the display, it may not be obvious which window will receive the path name that PAM will send.

To select a destination window for *send*:

1. Select the desired destination window.
2. Select the PAM window.
3. Press Send (f3).

set_font

Name

set_font—set the character font in a window

Syntax

set_font *font_file_name window_name*

Location

Standard Applications disc, in **/apps/fonts**

Description

The *set_font* program enables you to change the font (or character set) of an alpha window. Running the program causes the specified font file to be loaded into system memory and to be used for the character set in the specified window. With the *fedit* program, *set_font* enables you to create and access customized character sets.

A number of ready-made fonts are included on the Standard Applications disc, in the **/apps/fonts** folder.

Try this:

1. Insert the Standard Applications disc in the internal disc drive and open the **/apps/fonts** folder.
2. Enter the *echo* command in the PAM command line to create a window named **test**. Type: `echo #test` (Return).
3. Save the **test** window with Save (key (f4) in the System menu).
4. Load a new font into the **test** window. Type in the PAM command line: `set_font block12x16 test` (Return). (Note the omission of the # symbol.)
5. When the *set_font* window reports that the specified font has been set, select the **test** window and type away!

Some application programs, such as HP Calculator, have their own fonts and load them into system memory when first started. A font will stay in system memory until no window is using the font any longer.

The *system font* resides permanently in the ROM disc and is loaded during the power up procedure for the computer. Its path name is **/rom/font** and the size of its character cells is 6 dots by 8 dots.

To establish a new font for subsequent windows, change the font in the *PAM* window. Each new program started from PAM will inherit the current PAM font (unless the program loads its own font).

Examples

```
set_font cour9x15 MyWindow
set_font cour9x15 PAM
```

Limitation

The maximum size of any font is 16 dots by 16 dots per character cell.

See Also

fedit

status

Name

status—report status information

Syntax

status
status -help

Location

Utilities disc

Description

Status creates a window in which three system resources are inventoried:

- The amount of available *system memory* (in kilobytes).
- The names, drive names, and available storage (in kilobytes) of all *discs* currently in the system.
- The names, identification numbers, window names, and running times (in elapsed *minutes* and *seconds*) of all *programs* currently executing.

Use *status* when you want to check:

- How much memory a particular program requires (run *status* before running the other program, then run *status* while the other program is running, then take the difference in reported system memory).
- How much space you have left on a particular disc.
- How many programs you have running (if you're not certain that all program windows are visible).

Example

```
status
```

Limitation

Status can take a long time to complete, particularly if you are using a variety of disc devices in your system.

See Also

terminate

stopprint

Name

stopprint—stop printing the current file

Syntax

stopprint

Location

Built into PAM

Description

The *stopprint* command causes the current printer device to stop printing a file. Use *stopprint* after you've issued a PAM *print* command in order to cancel the print operation.

Example

```
stopprint
```

Limitations

- The *stopprint* command affects only *one* file—the file currently being printed. If you've started printing a number of files, then you must issue a series of *stopprint* commands to cancel the print operation for all the files in the list of files waiting to be printed.
- The internal printer maintains a 1K *print buffer* in which characters are stored before they're printed. After entering a *stopprint* command, you'll need to wait until the printer buffer is emptied (of up to 1,024 characters) before the printer actually stops printing.

See Also

print

S Chapter 3 of this guide

Name

terminate—terminate the execution of a program

Syntax

terminate *program_number*
terminate -help

Location

Utilities disc

Description

Terminate causes:

- The program specified by the *program_number* to be eliminated from memory.
- All resources used by the specified program to be reclaimed by the system.

Note that *terminate* may not eliminate the *window* in which the specified program runs. However, the window of the terminated program will be eliminated the next time the computer begins executing another program.

Normally, you won't need to use the *terminate* program—you can eliminate virtually any program using the Exit function key for that program, or else the (Stop) key on the keyboard, or else the (Shift)(Stop) keystroke.

In those situations when you want to use *terminate*, run the *status* program first to determine the *program_number* of the desired program. Then include the *program_number* after typing `terminate` in the PAM command line.

Examples

```
terminate 32
```

See Also

status

time_of_day

Name

time_of_day—display a clock window
—set the system clock

Syntax

time_of_day
time_of_day *hour:minute*
time_of_day *hour:minute month/day/year*
time_of_day -help

Location

Utilities disc

Description

By itself, *time_of_day* causes a small digital clock window to appear in the upper right corner of the display. The clock will be updated every minute.

If you specify the *hour:minute* parameters, the clock will be set to that time—the date will not be affected. Options:

- *hour:minute*—sets the clock according to 24-hour notation.
- *hour*:minute **am**—sets the clock to *am* time.
- *hour:minute* **pm**—sets the clock to *pm* time.

To set the *date,* include the *month*/*day*/*year* parameters after the *hour*:*minute* parameters.

Note the time displayed in the clock window is dependent on the TZ (time zone) variable in the current PAM environment. If no environment file has been read since you've powered up the computer, then the time zone variable will default to Eastern Standard/Eastern Daylight Savings time.

The reading in the clock window will automatically convert to and from local standard and daylight time during the year.

Examples

```
time_of_day
time_of_day 9:30
time_of_day 3:30 pm
time_of_day 15:30  10/18/1985
time_of_day 3:30 pm 10/18/1985
```

See Also

getenv

unmount_disc

Name

unmount_disc—unmount a disc from the file system

Syntax

unmount_disc *drive_name*
unmount_disc -help

Location

Utilities disc

Description

The *unmount_disc* program enables you to remove (or "unmount") flexible discs from external disc drives.

Do not remove a flexible disc from an external disc drive without first executing the *unmount_disc* program. Otherwise, the integrity of the data on the flexible disc may be destroyed.

Running the *unmount_disc* program ensures that all information shared between system memory and the disc is up-to-date and that no program is currently using any of the files on the disc.

Notes on *unmount_disc*.

- You *don't* have to use *unmount_disc* to remove a disc from the internal disc drive.
- You won't be allowed to unmount a disc that has an *open file* on it. For example, if you're editing a file that resides on the flexible disc and try to unmount the disc, the *unmount_disc* program will report that the disc can't be properly removed while the file is still open. In such a case, close the file, or stop the program that is using the file, and try again.
- You won't be allowed to unmount a disc if the open folder is on that disc. In such a case, open a folder on another disc and then try again.

After you've unmounted one disc and inserted another flexible disc in its place, you should run the *mount_disc* program to add the new disc to your file system.

Examples

```
unmount_disc D000
unmount_disc D001
```

See Also

mount_disc
scan_discs

Limitation

You can unmount only one disc at a time.

verify_disc

Name

verify_disc—verify the information on a disc

Syntax

verify_disc
verify_disc *drive_name*
verify_disc -help

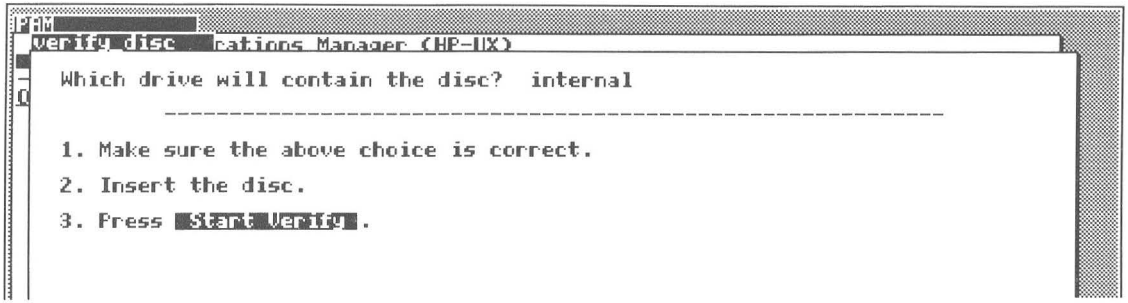
Location

Utilities disc

Description

The *verify_disc* program checks the integrity of the file system on a specified disc, attempts to locate misplaced files, and displays statistics on the number of files and folders on the disc as well as the amount of free space on the disc.

If you start *verify_disc* without specifying any parameters, you'll create the following program window:


PAM
verify_disc
cations Manager (HP-UX)
Which drive will contain the disc? internal
1. Make sure the above choice is correct.
2. Insert the disc.
3. Press Start Verify .

Note that the *verify_disc* window is ready to verify the disc in the internal disc drive.

If you specify a *drive_name* before running the program, then the `drive` field will be set to that value when the program starts.

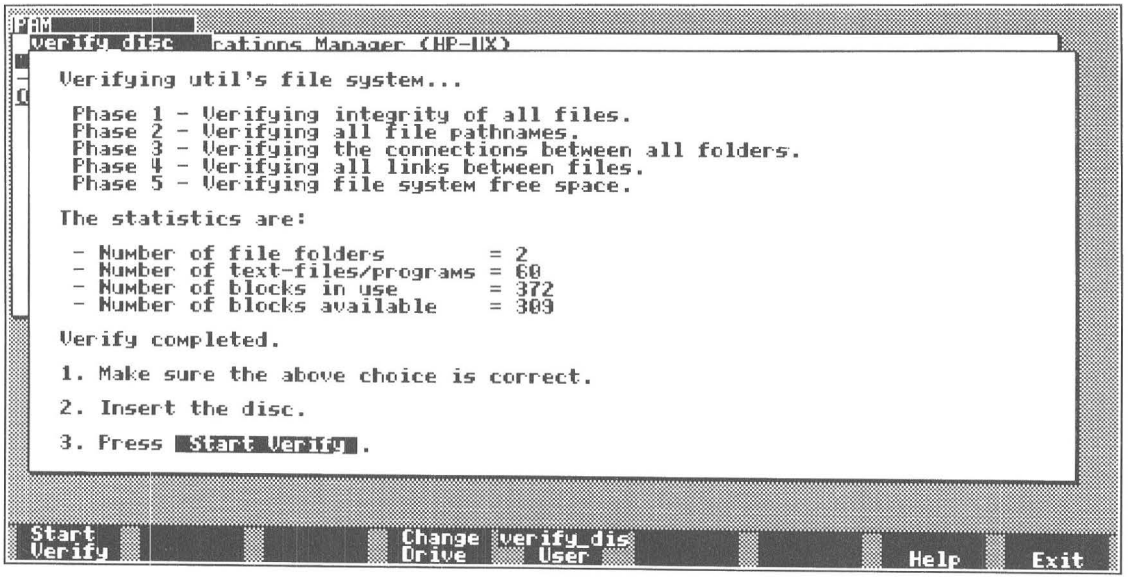
The following function keys control the *verify_disc* program:

Function Keys for Verify_disc

Function key	Description
Start Verify (f1)	Causes the program to examine the disc in the specified drive.
Change Drive (f4)	Enables you to specify a different *drive_name* where the disc is inserted. Pressing the key moves the cursor to the `disc drive` field and clears that field. Type in the name of the disc drive, such as `D000`, and press Return. To restore the field to its previous value, backspace over the field and press Return when the field is blank.
Help (f7)	Creates a "help" window, separate from the *verify_disc* window, in which you can read usage information about the program.
Exit (f8)	Ends the program, cancels the current verify operation (if in progress), and eliminates the *verify_disc* program window.

...verify_disc

Here is a typical sample of output after *verify_disc* has completed verifying a disc in the internal disc drive.


PAM
verify_disc
cations Manager (HP-UX)
Verifying util's file system...
Phase 1 - Verifying integrity of all files.
Phase 2 - Verifying all file pathnames.
Phase 3 - Verifying the connections between all folders.
Phase 4 - Verifying all links between files.
Phase 5 - Verifying file system free space.
The statistics are:
- Number of file folders = 2
- Number of text-files/programs = 60
- Number of blocks in use = 372
- Number of blocks available = 309
Verify completed.
1. Make sure the above choice is correct.
2. Insert the disc.
3. Press Start Verify .
Start Verify
Change Drive
verify_dis User
Help
Exit

The amount of used and available disc space is measured in *blocks*, equal to 1,024 bytes.

After completing a verify operation, *verify_disc* will display a message: `Verify completed.` *The program will continue executing.* To examine another disc, follow steps 1 through 3 in the *verify_disc* window. To end the program, press **Exit** (f8).

If, for some reason, *verify_disc* finds a file that is not correctly associated with any folder on the disc, then the program will attempt to recover the file. *Verify_disc* will:

- *Rename* the file with a numeric code.
- *Move* the file to the outermost folder on the disc.

Afterwards, you can determine the identities of any "repaired" files and move the files back to their original folders.

Examples

```
verify_disc
verify_disc D000
```

Limitations

- The (Back space) key is the only editing key that you can use to edit any of the fields in the *verify_disc* window.
- The execution time of *verify_disc* can be lengthy, especially for hard discs.

view

Name

view—view the contents of a file

Syntax

view *file_name ...*
view *file_name ...* #*window_name*

Location

Built into PAM

Description

The *view* command enables you to list the contents of text files in a special *view window* created by PAM. The effect is similar to highlighting the *file_name* of the desired file and pressing View (f1).

The view window is associated with two function keys:

Function Keys for the View Window

Function Key	Description
More (f1)	Causes PAM to display the next windowful of text.
Quit (f8)	Causes PAM to eliminate the view window.

Note that pressing *space bar* has the same effect as pressing More (f1). Also, pressing the Q key (for lowercase **q**) has the same effect as pressing Quit (f8).

The view window will remain in the display until you eliminate it with Quit, Q, or Stop.

Using *view* is a convenient way of displaying the contents of *script files*. Script files appear in the PAM window as *program files*, but actually consist of lines of text.

You can specify a *window_name* parameter after the *view* command name in order to name the resulting *view* window.

You can also "pipe" the output of one program into the *view* command using the vertical bar (|) symbol in the PAM command line. For example, entering `/hp-ux/ls -al | view` causes the *ls* program on the **/hp-ux** disc to direct its output to a *view* window.

Note that you can list other types of files, including folders, but the displayed information will probably appear as a jumble of characters.

Examples

```
view Memo-3-31
view One Two ../Three
view ThisFile #ThatWindow
ls -al | view
```

Limitation

Entering *view* by itself causes an error message.

See Also

Chapter 3 of this guide
Chapter 4 of the tutor or the user's guide

5 Installing and Using Peripherals

Introduction

Although you don't need peripherals to operate the computer, they are useful in many situations. For example, you might want to add a disc drive for additional mass storage, use a letter-quality printer to obtain printed copies of documents, or use a plotter to make copies of graphs you created on the computer.

This chapter provides information for connecting peripheral devices to the computer and using those devices. In this chapter, you'll learn how to use the built-in HP-IB interface and a serial interface module.

Steps to Using Peripherals

Before peripherals can be accessed, they must first be connected to the computer with an HP-IB cable or an interface module and the proper cable. After a peripheral is properly connected, it might be necessary to install a *driver program* to make the device accessible through the file system.

To use peripherals with the computer, they must be set up according to the following steps:

1. **Connect the peripherals to the computer.** You can connect peripherals to the built-in HP-IB interface, or to another interface that plugs into one of the I/O ports on the back of the computer. This manual describes how to connect and use peripherals with the serial interface card.
2. **Install one or more drivers, as needed.** A *driver* is a program that you use to operate an interface or to access a device. When you connect a peripheral device, you might need to *install* one or more device drivers (that is load them into the computer) before you can use the peripheral.
3. **Access the peripherals.** Peripherals are typically accessed like files. You read from them and write to them.

Example: You have an HP 9133D combination hard disc/microfloppy disc drive that you want to connect to the computer and use. The disc drive contains a hard disc and a slot for a microfloppy disc. The disc drive has an HP-IB interface that you can connect to the computer. To connect the disc drive:

1. Check the HP-IB address of the drive. The disc drive has a preset HP-IB address of 0. If no other HP-IB peripherals have this address, you can go on to the next step. Otherwise, you might need to change the address setting of the disc drive. (HP-IB address settings are described later in this chapter.)
2. Using a right-angle HP-IB cable, connect the disc drive to the computer.

3. Insert a formatted disc into the peripheral disc drive and switch it on. The driver for disc drives (drivers are described later), is already installed in the system.
4. Switch on the computer. After a few moments, the computer displays the contents of the root folder. You can now access files on the disc drive.

Example: You have an HP Laserjet Printer that you want to connect to the computer and use. The printer uses a serial interface. Connecting it to The Integral PC requires an HP 82947D serial interface cable and an HP 82919A Serial Interface. To use the Laserjet printer:

1. Connect the Laserjet printer to the computer using the serial interface cable and the serial interface module. (Refer to the user's guide for the interface module for information about installing the module in the computer.)
2. Switch on the Laserjet printer. The ready light will blink for about a minute until it is completely warmed up. Then the ready light will be steadily on, indicating that the Laserjet is ready to use.
3. With the system disc in a disc drive connected to the system and the root folder open, type

   ```
   /system/load_serial
   ```

   and press (Return). This loads the *driver*, which controls the flow of information through the serial interface.
4. In the command line, type

   ```
   printer_is cul00 9600 xon-xoff
   ```

   and press (Return). (This assumes that the printer is connected to the first serial interface on the computer.)
5. Use the *print* command to send information to the printer.

Connecting Peripherals

The HP Integral Personal Computer has a built-in HP-IB interface for connecting peripherals. You can also install several types of interfaces in the I/O port on the back of the computer.

HP-IB Peripherals

The computer has a built-in HP-IB interface that allows it to access HP peripherals that conform to the IEEE-488-1978* interface standard. The purpose of HP-IB is to provide for mechanical, electrical, timing, and data compatibilities between all devices adhering to the standard.

Peripheral Address Settings. Some common HP-IB peripherals have a preset address. Typically, Hewlett-Packard disc drives have the preset address of 0, printers have the preset address of 1, and plotters have a preset address of 5. It is not necessary to change the HP-IB address of a peripheral unless duplicate addresses exist on the other peripherals to be connected. If the peripheral address needs to be changed, consult the owner's manual supplied with the peripheral for information on changing its address.

HP-IB Cables. The HP-IB connector on the back of the computer is used for an HP-IB cable that allows the computer to access HP-IB peripherals. Because the computer has a vertical HP-IB connector, you should use a right-angle cable. The end of the cable has a piggy-back connector that connects to a peripheral. Any peripheral that does not directly accept the cable connector requires an adapter. The following accessories are available for the HP-IB interface:

- HP 92220R 0.3-Meter Right-Angle Cable.
- HP 82977A 1-Meter Right-Angle Cable.
- HP 82977B 2-Meter Right-Angle Cable.

* Institute of Electrical and Electronic Engineers 448-1978 standard for a general-purpose interfacing bus.

Do not connect more than three HP-IB cable connectors to one HP-IB receptacle on the computer or a peripheral. Doing so could exert enough force on the connector mounting to cause mechanical damage. Other peripherals can be added to the interface by stacking the cable connectors on peripherals (instead of the computer).

HP-IB Cable Length Restrictions. To insure proper operation of the interface, the total length of HP-IB cables connected to the integrated HP-IB interface is restricted. The following two rules should be observed:

- The total length of cable permitted must be less than or equal to 2 meters times the number of devices connected together.
- The total length of cable must not exceed 20 meters.

For example, there may be up to 4 meters of cable between the first devices (2 devices $\times$ 2 {meters/device} = 4 meters). Additional devices may be added using 2-meter cables up to a total of 10 devices (10 devices $\times$ 2 {meters/device} = 20 meters). Alternatively, one 4-meter and eight 2-meter cables could be used (4 + (8 $\times$ 2) = 20). If more than 10 devices are to be connected together, cables shorter than 2 meters must be used between some of the devices. Fifteen devices, for example, can be connected together using one 2-meter cable and thirteen 1- meter cables (2 + (13 $\times$ 1) = 15). Other combinations may be used as long as both rules are observed.

HP-IB Metric Conversion Kit. The interface cable is supplied with mounting fasteners having metric threads. Other HP-IB instruments, however, may have either National Coarse (American) threads or metric threads. The American-threaded fasteners are chromium plated, while the metric-threaded fasteners are black.

Since metric and American threads cannot be connected together, a conversion kit is available. Use this kit to replace the mounting fasteners on any HP-IB cable connector. Metric conversion kits can be ordered by specifying HP part number 5060-0138.

Connecting HP-IB Peripherals. Make sure that you read and understand the procedure presented in this section before you connect a peripheral to the computer. Each peripheral comes with its own manual, and you should refer to the individual owner's manual for complete installation instructions. The following procedure applies only to disc drives, printers, and plotters available from Hewlett-Packard.

To avoid personal injury and equipment damage, do not deviate from the order of the following steps when installing peripherals.

1. Make sure that the power cord is plugged into a grounded (three-wire) ac outlet and the computer is switched *off*.
2. Make sure that the power switches on all peripherals to be connected to the HP-IB interface cable are in the off position.
3. Check the address switches of the peripherals to ensure that they have unique address settings.
4. Connect the right-angle end of the interface cable to the computer and the other end to the first peripheral. If other peripherals are to be connected at this time, install the cable connectors to the remaining peripherals in a piggy-back fashion. Use the accessory cables listed earlier in this chapter and be sure to observe the cable length restrictions on the previous page.
5. Plug a power cord into the receptacle on the back of each peripheral and the other end of the cord into the wall outlet.

Do not use extension cords under any circumstances. Such use may result in data errors and increase the risk of safety hazards. If you wish to use a multiple outlet strip to plug in the components of your system, you must use one that utilizes grounded three-prong outlets and incorporates a circuit breaker.

6. After all peripherals are connected to the HP-IB interface (refer to the owner's documantation for each device for information about connecting the device to HP-IB), switch on all peripherals to be operational on the interface, and then switch on the computer. A peripheral that is connected to the interface can have its power off without affecting interface operations, as long as more than 50 percent of the peripherals are switched on. For example, if there are three peripherals, two must have their power on; if there are two peripherals, both must have their power on.

Removing HP-IB Peripherals. Use the following steps in the order given to remove peripherals from the HP-IB interface.

1. Switch off all the peripherals connected to the HP-IB interface.
2. Switch off the computer.
3. Disconnect the HP-IB interface cable from each peripheral you intend to remove from the interface. If all peripherals are not to be removed from the interface, you may return power to the remaining peripherals after the desired peripherals have been disconnected.
4. Disconnect the interface cable from the computer (if you desire to do so).

Serial Peripherals

You might find that you want to connect peripherals to the computer using a serial interface. The computer enables you to connect up to 10 devices using serial interfaces.
To use a serial device, you must have an HP 82919A Serial Interface module. (Refer to the *HP 82919A Serial Interface Owner's Manual* for information about installing and using the serial interface.) A bus expander enables you to connect up to 10 devices to the computer. You must have one serial interface for each device you connect to the computer. After you connect a device to the computer using the serial interface, you need to load the serial driver, and possibly a driver for that device, to operate the device.

Installing a Driver

A *driver* is program that controls the operation of a device (such as a disc drive) or an interface (such as HP-IB). The operating system uses drivers for accessing peripherals.

A separate driver is used for each interface that peripherals are accessed with. A driver for the built-in HP-IB interface is included in the standard system. Access to peripherals connected to the computer with other interface modules requires the use of additional drivers.

The HP-IB interface driver can control up to 15 peripherals connected to the built-in HP-IB interface.

When additional interface drivers are installed for use with other interfaces, separate files are created in the **dev** folder for directly accessing the peripherals connected to each additional interface module.

Interface drivers are necessary to access all peripherals connected to the computer. The standard system includes interface drivers for the built-in HP-IB interface. The use of other interfaces requires the installation of additional drivers. The actual installation of an interface driver is performed by a utility. Utilities for installing interface drivers are available from Hewlett-Packard. Drivers for the following interfaces are also available from Hewlett-Packard:

- HP 82919A Serial Interface.
- HP 82920A Current Loop Interface.
- HP 82921A 300/1200 BPS Modem.

Driver Installation Procedure. Interface drivers are normally installed at the beginning of each session by running a utility. The utilities make it possible to access a specific interface by creating a file in the **/dev** folder. When output is directed to this device file, the peripheral connected to a specific interface module is accessed. The following general procedure is used:

1. Install the interface module in the computer. (Refer to the owner's manual of the interface for information about installing and using that interface.)
2. Connect the interface module to the peripheral with the proper cable.
3. Switch on the peripheral connected to the interface module and any other peripherals to be used in the current session.
4. Switch on the computer.
5. Run the utility to install the appropriate drivers.
6. Use the device file created by the utility for accessing the peripheral.

The Serial Interface Driver. The serial interface driver enables you to use peripherals connected to the computer using the serial interface. The driver controls how information is passed through the interface. The serial driver is located on the *system* disc included with the computer. To load the driver, insert the disc in the internal disc drive (or in a peripheral disc drive using the procedures described in this chapter), ensure that the disc's folder is open, and type:

```
load_serial (Return)
```

If the device is the first serial device connected to the computer, its file name is **cul00**. If it is the second serial device, the file is **cul01**, and so on up to **cul09**. This is the file name you would use, for example, in a *printer_is* or a *plotter_is* command.

A convenient method for installing frequently-used interface drivers is using a script file that contains all the commands you would normally enter in the command line to install a specific interface driver. Copies of the utilities needed to install the interface drivers can be stored on the same disc as the script file. The interface driver can then be installed by simply entering the name of the script file as a command. If you name the script file **Autost**, it can be executed automatically when the computer is switched on. Refer to "Specifying Operations With Files" in chapter 3 for information about using script files and **Autost** files.

Using Peripherals

Peripherals are accessed with device files, like internal devices, but usually need to be assigned to a device file by running a utility before they can be accessed. HP-IB devices are identified by their HP-IB address. Devices connected to serial interface modules are identified by the device files the system creates for each serial interface module installed in the computer.

The next few sections explain when it is necessary to use utilities to access devices and provide other information about using specific peripheral devices. Accessing disc drives, printers, and plotters with the HP-IB interface or a serial interface module is covered. Information about using device drivers to access other specialized devices and interfaces is provided later in this chapter.

The dev Folder

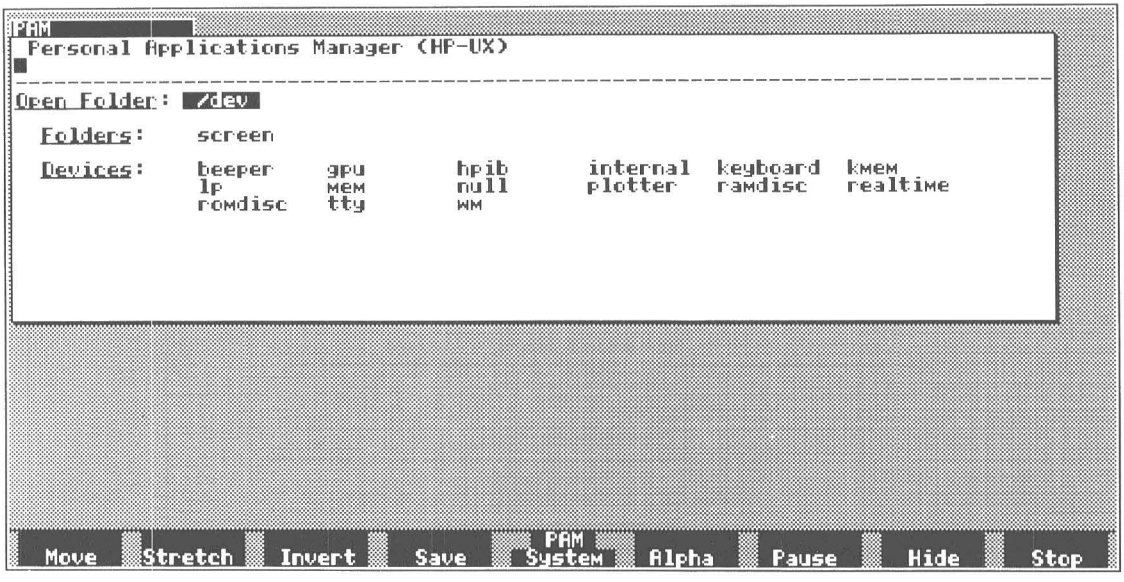
The **dev** folder is where all the device files supplied with the computer are stored. Device files are used for accessing devices with input and output operations. Reading from a device file enables you to input information from a device. Writing to a device file causes data to be output on a device.

The name of a specific device file is used to identify the device it provides access to. When used as a parameter in commands, a device file is used as a name for the device it currently accesses.

Several of the device files in **dev** can be used as parameters for commands and utilities. The device files that can be used as parameters include the following:

- The **internal** file can be used to identify the internal disc drive as a parameter with utilities for disc maintenance and other disc operations. The internal drive is accessed by default when a disc drive is not specified.
- The **lp** file can be used for sending output to the internal printer or to certain peripheral printers. The printer accessed by this file is called the *active printer*.
- The **plotter** file is used for accessing plotter windows and compatible peripheral plotters. The plotter accessed by this file is called the *active plotter*.

The following illustration shows the devices files included with every HP Integral Personal Computer. (Your computer might have more device files than shown here.)


PAM
Personal Applications Manager (HP-UX)
Open Folder: /dev
Folders: screen
Devices: beeper gpu hpib internal keyboard kmem
lp mem null plotter ramdisc realtime
romdisc tty wm
Move Stretch Invert Save PAM System Alpha Pause Hide Stop

Peripheral Disc Drives

You might want to use peripheral disc drives to provide additional mass storage or to make it easier to use different discs. When using peripheral disc drives, you must be careful to follow the correct procedure for inserting and removing discs so that the computer always knows which discs are available to it.

Before inserting or removing a disc in a peripheral disc drive, you must inform the computer of your intention. You can do this using the *mount_disc* and *unmount_disc* utilities. (These utilities are described in chapter 4.) If you don't follow the correct procedure for inserting or removing discs from peripheral disc drives, you might lose some or all of the data on a disc.

The following is the correct procedure for inserting and removing discs from peripheral disc drives:

Inserting and Removing a Disc. You might want to insert discs in peripheral disc drives before you turn the computer on or you might want to exchange discs while the computer is switched on.

To insert a disc before the computer turns on:

1. Make sure the computer, the external disc drive, and all other peripherals are turned off.
2. Insert your disc (or discs) in the external drive and switch on the drive.
3. After the disc drives and all peripherals have completed their power-up sequences, switch on the computer and wait for the computer to display folder information. The computer automatically runs the *scan_discs* utility (described in chapter 4), which checks for new disc drives and puts the drive names in the **dev** folder. When the computer is turned on, it automatically mounts all discs in all connected, powered-up disc drives.
4. If you switch on the computer and then want to switch on a disc drive, run the *scan_discs* utility to register the new drive and mount the discs in the drive.

To exchange discs while the computer is switched on:

1. Before removing a disc from an external drive, use the *unmount_disc* utility to remove the disc file system from the root folder (*/*). (*Unmount_disc* is described in chapter 4.)
2. Remove the disc from the drive.
3. Insert another disc into the drive.
4. Use the *mount_disc* utility (described in chapter 4) to put the disc file system into the root folder.

Remember that *scan_discs* is used when you add a disc drive to the system, and *mount_disc* and *unmount_disc* are used for inserting and removing discs from drives that have names in the **dev** folder.

Accessing Disc Files. Disc files are accessed through folders in the file system. The *scan_discs* utility is run automatically at power on to mount any discs in peripheral drives connected to the integrated HP-IB interface. Mounting is an operation the system performs that places the uppermost folder of a disc in the root folder of the file system with a folder name that corresponds to the disc name. Unmounting makes a disc inaccessible to the computer by removing its uppermost folder from the root folder of the file system.

The uppermost folder of a disc is the same as the volume label of that disc.

Operations that involve accessing files and folders on a disc in a peripheral drive are performed in the same manner as operations with files stored on the disc in the internal drive. When certain operations are performed on a disc, such as initializing or formatting a disc with the *format_disc* utility, the device name of its drive is used as a parameter instead of the names of folders or files stored on the disc.

Do not use double-sided discs with a single-sided peripheral disc drive. The additional files and system data stored on the reverse side of a double-sided disc cannot be accessed by a single-sided drive. The absence of certain data about the file system may cause unpredictable results when files are accessed. You can use single-sided discs in either drive, but the amount of data that can be stored on them is much less than can be stored on a double-sided disc.

Disc Drive Addresses. Peripheral disc drives are identified in certain commands and utilities with device names. A *drive name* is a device name that identifies a disc drive. Drive names are actually the names of device files created by the system in the **dev** folder when the computer is turned on. A drive name has the following form and uniquely identifies a peripheral drive:

□ *volume—number* *HP-IB—address* *unit—number*

The drive name begins with the letter □ and is followed by three digits. The information these digits represent is determined at power on by the system for each disc drive accessible through the HP-IB interface.

- The first digit is the volume number of the disc. This is 0 for most flexible discs and most hard discs.
- The second digit is the HP-IB address of the disc drive. Disc drives have a preset address of 0. Some hard disc drives that include a flexible disc drive use two separate HP-IB addresses, so the system will create two device files for accessing these disc drives.
- The third digit is the unit number. This is printed on the disc drive unit, beside the slot where the disc is inserted. (It is usually 0 or 1.) The system creates separate device files for units 0 and 1.

Examples. The drive name `D000` identifies unit 0 of a peripheral disc drive with an HP-IB address of 0 and a volume number of 0. The drive name for unit 1 on the same disc drive would be `D001`. The drive name `D170` identifies unit 0 of a drive at address 7. The volume number of the drive is 1.

If your system includes a hard disc drive, often one device file is created for accessing it. However, some hard disc drives can have more than one drive name assigned. You can set some hard disc drives to have more than one *volume*. A volume is simply a portion of the hard disc memory that is treated as a single disc.

Examples. You have an HP 9133D combination hard disc drive/floppy disc drive connected to your computer. (No other peripheral drives are connected.) The HP-IB address of the disc drive is 4. The drive has a switch on it that enables you to specify the number of volumes on the hard disc. If you select three volumes, then you have the following drive names:

Drive Name	Description
`D041`	Specifies the floppy disc drive (unit 1), volume 0.
`D040`	Specifies the hard disc drive (unit 0), volume 0.
`D140`	Specifies the hard disc drive (unit 0), volume 1.
`D240`	Specifies the hard disc drive (unit 0), volume 2.

When determining what device file to enter as the device name for a command or utility, the **dev** folder can be opened to display the current device files. If a device file for a peripheral drive is not present in the **dev** folder, then it cannot be accessed. If this occurs, make sure the HP-IB address of the peripheral drive is set correctly and that it is properly connected to the computer.

Examples. The following commands use device name parameters to specify operations on discs in peripheral disc drives.

```
verify_disc D000
```

Checks the file on the disc in unit 0 of a peripheral disc drive.

```
format_disc D001
```

Allows you to initialize the disc in unit 1 of a peripheral disc drive.

```
rename_disc zenon D010
```

Changes the disc name of a disc drive.

```
copy_disc internal D000
```

Copies the file system on the disc in the internal drive to the disc in unit 0 of the peripheral disc drive.

```
copy_disc D000 D001
```

Copies the file system from the disc in unit 0 of a peripheral disc drive unit to the disc in unit 1 of the same drive.

Peripheral Printers

Peripheral printers can be used with the computer for special applications that the internal printer is not suited for. A printer is usually connected to the computer with the built-in HP-IB interface or with an optional serial interface card. The *printer_is* utility is used to change the printer used by the system; that printer becomes the *active printer*. The utility can be run several times during a session to select a different peripheral printer or to select the internal printer again. The *printer_is* utility is stored on the utilities disc provided with your computer. This disc can be inserted in the internal disc drive for accessing the utility from the PAM window or copying the utility to the root folder for use later in a session. (The *printer_is* utility is described in more detail in chapter 4.)

To specify a peripheral printer as the active printer, enter its HP-IB address or its RS-232 device name as a parameter with one of the following commands:

```
printer_is HP-IB—address
printer_is RS-232—device—name
```

Other optional parameters for this utility are described in chapter 4. To access a peripheral printer:

1. Ensure that the computer and all peripherals are connected and turned on.
2. Run the *printer_is* utility to select the peripheral printer as the active printer. The file **lp** can now be used to access the printer.
3. Use the *print* command or redirect output to **lp** to specify files for printing. After running the *printer_is* utility, the specified printer is accessed automatically.

Depending on the interface used for connecting a peripheral printer to the computer, different actions are taken by the computer at power on. Printers connected to the built-in HP-IB interface become accessible with the HP-IB address currently set on the printer. To access an HP-IB printer with an address of 1, the command is:

```
printer_is 1
```

When the peripheral printer you are using is connected to the computer with a serial interface module, the system creates a device file in the **dev** folder for accessing the printer at power on. If only one serial interface module is used, then the file **cul00** is created. To access a printer connected to the first or only serial interface module installed in the computer, the command used is:

```
printer_is cul00
```

If multiple serial interface modules are installed on the computer, a separate device file is created for each module at power on. The files are numbered from **cul00** through **cul09.**

These files are ordered relative to the ports on the rear panel of the computer. Beginning with the port A, the first serial interface card is accessed with the file **cul00** A peripheral printer connected to an additional serial interface module in port B would be made the active printer by specifying the file **cul01,** as follows:

```
printer_is cul01
```

To select the internal printer as the active printer again, enter `internal` as a parameter, as follows:

```
printer_is internal
```

Note that the internal printer is used for printing the screen contents when (Shift)(Print) is pressed even if a peripheral printer is the active printer.

Examples. The following commands cause different printers to be selected as the active printer. Files to be printed need to be submitted to a specific printer while it is the active printer.

```
printer_is 2
```

Selects an HP-IB printer with an address of 2 as the active printer.

```
printer_is internal
```

Selects the internal printer as the active printer.

```
printer_is cul00
```

Selects the printer connected to the first serial interface module as the active printer.

If you want to use a specific printer with a program, select that printer as the active printer before running the program. A program can use only the printer that is active at the time the program starts.

Peripheral Plotters

Peripheral plotters are accessed with device files, like other peripherals. Before a plotter can be accessed, the *plotter_is* utility must be run to specify an active plotter device. Note that the system does not assign a default active plotter. After specifying an active plotter, it can be accessed with the device file **plotter.** The following procedure is used for accessing peripheral plotters:

1. Ensure the computer and all peripherals are connected and switched on.
2. Run the *plotter_is* utility to make the peripheral plotter the active plotter.
3. If you use a file containing plotter instructions, make sure it can be accessed through the file system. Send output from the file containing plotter instructions to `plotter`.

After the *plotter_is* utility is run to change the active plotter, the device file **plotter** is used for accessing the current active plotter. Refer to chapter 3 for information about redirecting output.

A peripheral plotter can be connected to the HP-IB interface or a serial interface card as the active plotter with the following commands:

`plotter_is` *HP-IB—address*
`plotter_is` *RS-232—device name*

You can specify optional parameters, as described in chapter 4. For HP-IB plotters, the HP-IB address is used for changing the active plotter. HP-IB plotters have a preset address of 5. To access an HP-IB plotter with an address of 5, the following command is used:

```
plotter_is 5
```

Plotters connected to the computer with a serial interface are identified with device files numbered from **cul00** through **cul09** when the active plotter is changed. One of these device files is created for each serial interface module installed on the computer. The ordering of the files corresponds to the ordering of the interface cards in the ports on the rear panel of the computer. To access a plotter connected to the first or only serial interface module installed in the computer, the command is:

```
plotter_is cul00
```

If the device name entered does not identify a file in the **dev** folder, then a graphics window with the name you enter as a parameter is created. If a plotter window with the name entered already exists, then this window is selected as the active plotter.

Examples. The following commands cause different plotters to be made the active plotter.

```
plotter_is 5
```

Selects an HP-IB plotter with an address of 5 as the active plotter.

```
plotter_is cul00
```

Selects a plotter connected to the first serial interface module as the active plotter.

A Glossary

A

absolute path name: A list of folder names that specifies the location of a folder or file in relation to the root folder. The list starts with the root folder (*/*) and ends with the file or folder being identified. One slash separates each subsequent folder in the list. An absolute path name that identifies a file is also referred to as an absolute file name.

active window: The window that currently receives your input from the keyboard. The name of the active window is displayed above User or System in the center of the menu. This window is normally displayed in front of the other windows.

alpha window: A window used for displaying character data that is created by the PAM program automatically when programs, utilities, and other processes are started. A menu is available for using some of these features. The use of control codes and escape sequences is necessary for using the additional terminal features.

anchor: A movable bracket that is displayed over the upper left corner of a window. The bracket specifies the new location of the upper left corner of a window when it is moved. This corner is fixed and remains stationary when a window is stretched.

arrow keys: The ▲, ▼, ◄, and ► keys. These keys are used for moving the cursor, moving the graphics pen, rolling the contents of a window (with Shift), and moving the display pointer (with CTRL). Pressing two arrow keys such as ▲► specifies diagonal movement; in this case, up and to the right.

Autost file: A file name that is automatically searched for at the beginning of each session. If a file called `Autost` is in the top level folder of the disc in the internal drive at power on, it is accessed. The specific action depends on whether the file is a program, folder, or data file.

B

binary number: A number expressed in the base 2 number system. This system uses only the digits 0 and 1. In a computer, a binary 1 represents an ON state and a 0 represents an OFF state. The decimal values that can be represented with eight binary digits range from 0 through 255.

block device: A device that performs input or output operations with fixed units of data called blocks. The internal disc drive reads and writes blocks containing 1024 bytes each.

built-in device: A device contained in the computer that can be accessed without the use of an interface. The internal disc drive is an example of a built-in device.

buffer: Memory that functions as storage area for data that is not immediately operated on by the system. Buffers are used by devices and the computer whenever input or output operations are performed. When a data file is edited, for example, the edited version can be kept in a terminal buffer until the file is "saved" by storing it on a disc.

byte: The fundamental unit of information used by the computer. In data operations, each character is represented with one byte of memory. Each byte contains 8 binary digits or bits. A byte equates to a decimal number in the range 0 through 255.

C

character device: A device that performs input or output operations on a character-by-character basis. Most devices used with the computer besides disc drives are character-type devices.

character font: The specific bit image that is generated in a window when keys are pressed or output is directed to a window. The character set and display enhancements assigned to the character font determine what characters are generated. Two fonts, the base font and the alternate font, are available for each window. The base font is used unless the alternate font is selected.

character set: A specific sequence of characters and control codes used by the computer for handling information. Each character or code has a numeric value that establishes its order in the character set. The computer uses the ROMAN8 character set by default.

command: A sequence of characters that, when entered in the command line, cause an operation to occur. Commands generally start with a name that identifies an executable file, such as a program or utility, and may include parameters.

command line: The second line of the PAM window, where the cursor is located. Any characters you type in the PAM window go to the command line. When Return is pressed, the computer attempts to execute this text as a command.

compiler: A program that reads program statements from a data file and creates a program file containing instructions that cause the computer to perform the operations specified by the program statements. A different compilier is needed for each programming language.

computing environment: A group of settings the system uses for determining the result of specific operations. The environment is set by variables in an **.environ** file at power on and includes settings for the home folder and the search path, for example.

concurrent execution: The simultaneous execution of two or more programs, utilities, or commands in the computer. Commands entered in the command line are executed concurrently unless sequential processing is specified.

control code: A character that can cause a specific action to occur instead of displaying or printing a character. In the ROMAN8 character set, the decimal equivalents of 0 through 31, 127, 128 through 159, and 255 are reserved for control codes. Control codes can be entered by pressing CTRL and another key. Pressing CTRL J generates the line feed control code that moves the cursor to the next line, for example. In display functions mode, control codes are displayed as mnemonics. Control codes are also referred to as control characters. *See also* **control sequence** and **escape sequence**.

control sequence: A control code or escape sequence. *See* **control code** and **escape sequence**.

cursor: A position indicator that shows where the next character entered will be placed. In the PAM window, the cursor only appears in the command line. Unless the application limits cursor movement, the cursor can be moved to any character position with the arrow keys.

D

data file: A file that contains text or other data in a character format. Each byte in the file represents a specific character in a character set. Data files can be displayed with the **view** command.

data path: A stream of characters that is being transferred from one point to another point in the file system. A data path can result from input or output operations. Each program has three data paths associated with it: standard input, standard output, and error output. Each of these data paths can be directed to specific files, including device files, in the file system.

debugging: The examination of a program or file to determine if it will produce the desired results in different situations. The process of debugging can eliminate unexpected results that do not necessarily cause errors to occur.

decimal equivalent: The decimal value assigned to a character in a character set. In the ROMAN8 character set, the decimal equivalent of the line feed control code is 10. This value is used by the computer to represent each character stored in the computer and can be used to represent characters in a program.

default action: The action that occurs when a file is accessed. The file is usually accessed by entering its name as a command in the command line and pressing (Return). The system uses the file type to determine what action to take. Folders are opened, data files are displayed, and programs are executed.

default device: The device that is assigned to a device file at power on. The device file **lp** accesses the built-in printer. The default device for a specific device file can be changed by running a utility.

destination: In a data transfer operation, the file, folder, or disc to which data is transferred. In a copy operation, the file being copied to is referred to as the destination. When a compiler is used, the data file containing the compiled statements is called the destination file. *See also* **source**.

device address: A peripheral device setting that allows an interface in the computer to identify devices uniquely. The HP-IB interface can access each device it is connected to with a different address. The device address switch on a peripheral changes its address.

device file: A file that represents a specific device that the computer can access when performing input or output operations. The files **lp** and **plotter** are examples of device files that can be accessed for user output operations. Each device file contains information about what type of device it accesses and information for uniquely identifying the device it is currently assigned to. Device files are stored in the **dev** folder.

device name: The name assigned to each disc drive, plotter, printer, or other device by the operating system at power on, used as part of an HP-UX path name. This is a machine-dependent parameter. The path name identifies a specific device independently of the folders or files it currently provides access to in the file system. Device names may be used by single-user systems with removable file structures, such as discs. For example, a device name can be used to identify a disc drive as the source or target of a disc copy operation. The device names are listed in the **dev** directory.

directory: Another name for a folder. *See* **folder**.

display bar: An indicator used for selecting specific files or folders in the PAM window. The display bar can be moved over different file and folder names, including the name of the open folder, with (Tab) and (Shift)(Tab) or the arrow keys. Selecting a file or folder with the display bar and pressing (Select) is an alternative to entering its name in the command line.

display functions mode: A mode that causes control codes and escape sequences to be displayed instead of executed. This is useful for debugging purposes when output from a program, for example, is not what you expected.

display pointer: An image that can be moved anywhere on the screen to select a location. Depending on what is currently displayed in the location selected, different actions are taken. If the selected location is in a menu label, then the action of pressing the associated function key is performed when (Select) is pressed or when the left mouse button is pressed. The display pointer can be moved with a mouse or the arrow keys with (CTRL). The default pointer shape is an arrow, but different forms can be assigned programmatically. The display pointer is also called a *sprite*.

driver: A program which controls the input or output operations of an interface or a device. Different drivers are used for accessing different types of devices, such as printers or plotters. When a specialized device or interface is to be accessed, a loadable driver can be installed in the computer.

E

editing keys: Keys on the keyboard that can be used for updating information on the screen by deleting, clearing, and inserting characters. Keys are provided for editing on a character, line, or window level. Keys used for moving the cursor, rolling the window, and moving the display pointer are also considered editing keys.

environment file: A file containing variable assignments that set the computing environment. These include the HOME variable for naming the home folder and PATH variable for setting the search path. A file called **.environ** is accessed automatically after power on to set the computing environment. Other environment files can be used during a session to modify this environment.

escape sequence: A string of two or more characters that begins with the escape control character (ESC), and performs a specific function when received by a device, such as a printer or terminal. *See also* **control code** and **control sequence**.

extended character: A character accessed by pressing (Extend char) and another character key. Extended characters include foreign characters and other special characters in the second half of the ROMAN8 character set.

F

file name: A sequence of 1 to 14 characters used to identify a particular file. Any keyboard character can be used except slash and leading colon. File names are used in combination with folder names to form absolute path names and relative path names. The file name is entered into the directory in which the file is located. *See also* **absolute path name**, **file path name**, **relative path name**, and **simple file name**.

file path name: A file name preceded by a list of the successive folders that must be opened to access the file. For example, the file path name for a file named **Myfile** in a folder named **Myfold** on a disc named **Mydisc** would be `/Mydisc/Myfold/Myfile`. For further information, refer to the user's guide for your computer. *See also* **absoulte path name**, **file name**, **relative path name**, and **path name**.

file permissions: These determine what types of operations a file can be used for. There are three file permissions: *read* permission, *write* permission, and *execute* permission. Each permission is either set on or off for a file. File permissions can be changed by running a utility. Only files with execute permission can be executed as a command, for example. File permissions are also referred to as file modes.

file size: The amount of data that is stored in a file, represented in bytes. The size of a file determines how much space it uses when stored on a disc.

file system: The hierarchical structure of folders and subfolders that is used for storing and organizing information in the computer. The location of any file or folder in the file system can be expressed in relation to the root folder with an absolute path name.

file type: The file characteristic that is set when a file is created and determines how it is used. There are four file types: *data* files, *folder* files, *device* files, and *program* files. Data files can be changed to script files by modifying the file permissions.

folder: A file that contains pointers to other files or folders. These files or folders are what is logically stored in a folder file. Folder is another name for directory. *See also* **directory**.

formatting: The creation of a file system on a disc. This includes assigning the disc a volume name, calculating the number of files that can be stored on it, and creating a folder for storing that many files. Before a disc can be formatted, it must be initialized.

G

graphics window: A window used for plotting graphics data that is created when output is sent to the device file **plotter**. The plotting simulates the action of a mechanical plotter.

H

hidden file: A file or folder with a name that begins with a period. The period prevents the file name from being listed in the PAM window.

hidden window: A window that has been removed from the screen with the Hide function key, (f7) in the system menu. The banner of each hidden window is displayed in the lower left corner of the screen. Information in hidden windows is kept in off-screen storage until the window is displayed again by selecting its banner with the display pointer.

home folder: A folder identified by the computing environment. If this folder is included in the file system, it is displayed in the PAM window at power on and can be accessed during a session by executing the **cd** command without parameters.

HP-IB: The abbreviation for Hewlett-Packard Interface Bus. This interface standard allows multiple devices to be accessed with one interface. The integrated HP-IB interface on the rear of the computer can be used for accessing disc drives, printers, plotters, and other peripheral devices.

HP-UX: The Hewlett-Packard version of the UNIX operating system.

I

initialization: The preparation of a disc for storing information. The initialization process involves marking the disc medium with track and sector marks. After a disc is initialized, it can be formatted.

M

menu: A set of choices that are selected with the function keys, (f1) through (f8) (shifted and unshifted). Each choice is identified by a menu label. There are two menus, the system menu and the user menu. The system menu remains the same and can be accessed from any window. The user menu depends on what window is active and is different for different programs. The PAM user menu, for example, is different than the user menu displayed by a text editor.

menu label: An inverse block on the lower portion of the screen that corresponds to a specific function key. The menu labels can be turned on or off with (Menu). The eight menu labels correspond to the eight function keys. Menu labels that have a top and bottom half separated with a horizontal line represent two separate functions. The function on the upper half of the label is invoked by pressing (Shift) and the related function key.

mnemonic: Two or more characters that identify a control character, instruction, or any other phrase or name by using letters of the words it represents. The computer uses mnemonics for displaying control characters in display functions mode. The characters SO, displayed as ⟨ 5 ⟩ in display functions mode, are a mnemonic for the Shift Out control character.

mounted disc: A disc that is accessible through the file system. The folder with a name corresponding to the volume name of each mounted disc is stored in the root folder of the file system. When a file in a mounted disc is read or updated, the operating system automatically accesses the disc in the drive associated with the mounted disc.

mouse: A mouse is a hand-operated device used to move the display pointer and to select items located by the display pointer. By rolling the mouse on a table top, you can move the display pointer in any direction, select menu choices and windows, and turn the menu labels on or off. Many applications that are available for the computer incorporate the use of a mouse. We recommend the HP 46060 Mouse for use with the Integral Personal Computer.

N

null string: A representation of the absence of any characters. In a program character strings are usually enclosed in quotes. The null string is represented by two adjacent double quotes ("").

O

octal number: A number expressed in the base 8 number system. This system is often used for representing values stored in binary digits since groups of three binary digits can be conveniently converted to one octal digit. Octal numbers represent values with the digits 0 through 7.

open folder: The folder whose contents are currently listed in the PAM window. At any time, there is only one open folder. The open folder is also called the current folder.

operating system: The collection of programs used by the computer for managing its devices, processing capability, and memory.

overwriting: The process of replacing original data in a file with new data from an output operation. The data that is replaced is lost. Overwriting can occur accidentally when you intend to specify a new file for output but instead name a file that already exists.

P

PAM: The abbreviation for the Personal Applications Manager program. This program provides many of the features found in traditional UNIX shells and has a friendly user interface consisting of the PAM window and its user menu.

PAM command line: *See* **command line**.

PAM window: The window displayed by the PAM program. This window displays the open folder and a user menu for performing routine operations.

parameter: A data item that is placed after a command to specify how an operation is to be performed or to supply other needed information. Programs, script files, and other processes also use parameters. Parameters that are identified by their order of appearance in the command line are called positional parameters.

parent folder: The folder that a specified folder is stored in. A parent folder is usually identified in relation to the open folder. The parent folder of the open folder is the folder that is displayed when the open folder is closed. The name of the parent folder can be represented with two periods (. .) as a parameter.

path name: A list of file and/or folder (directory) names separated by one or more slashes, where each file is stored in the folder that precedes it. The last item in the path name may be either a folder or a file. Path names are used to specify folders or files that can be accessed through the file system. A leading slash represents the root folder. Example:
`/MEMOS/January/Re:Sales`.

peripheral device: An external device that is connected to the computer with an interface. A utility is normally run to associate a peripheral with a specific device file. The peripheral device can then be accessed by performing input or output operations with this device file. External disc drives, daisy-wheel printers, and mechanical plotters are examples of peripheral devices.

permission: *See* **file permissions**.

pipe: A data path that connects two programs that are executed concurrently. The pipe directs the standard output of one program to the standard input of the other program. A pipe between two programs is specified with a vertical bar (`|`) when the program names are entered in the command line. The use of pipes to connect programs is referred to as piping.

plotter device: A logical device that is associated with each graphics window. Each plotter device can access the plotter function keys and, when the graphics window is active, pen movement instructions directly from the keyboard.

print pitch: The number of characters printed per inch. The print pitch is used to represent a specific character size on a printer. By sending an escape sequence to a printer you can change the size of characters printed. The built-in printer has four print pitches.

printer waiting list: A list of one or more files that have been designated for printing while the printer is busy. These files are remembered by the system and are automatically printed when the printer stops. Files can be placed in the printer waiting list with the *print* command.

process group: A collection of programs that are concurrently executed and are associated with one application. The programs in a process group are typically controlled by one program called a parent process. If the parent process is stopped, other programs in the process group will also stop.

program file: A file that causes a process with the same name to begin in the computer when its name is entered in the command line. Program files are normally created with a compiler. Script files are special types of program files that are created by running a utility to make executable a data file that contains commands. The utility files provided with your system are also examples of program files.

programmatic access: The use of a program to perform operations with devices or files independently of user actions. Many of the capabilities of the computer can only be accessed programmatically because the necessary procedures involve complex statements that open files and perform input and output operations. These statements are not meaningful to the PAM program when entered in the command line.

R

RAM: The abbreviation for random access memory. RAM is the read/write memory resource of the computer that is used for running programs and storing needed information for current operations. Information that is accessed in each window is stored in RAM, for example. The amount of RAM available to the computer can be increased by installing plug-in memory modules.

RAM disc: A logical device that is used for storing files in RAM. The RAM disc folder is the root folder of the file system (/). Files on the RAM disc can be accessed very quickly.

redirection: The specifying of alternate files or devices for a program or other process to use for its input and output data paths. Certain symbols are used to identify files for standard input, standard output, and error output. A separate window can also be named for a program to use.

relative path name: A list of folder names separated by slashes that indicates the location of a specific file or folder in relation to the open folder. A relative path name that identifies a file is also referred to as a relative file name. *See also* **absolute path name**, **file name**, **file path name**, and **path name**.

reset: An operation that causes the computer to return to its power on state. A reset stops all programs and removes all windows.

root folder: The top-most folder in the file system. This folder is identified by a slash (/). All files and folders contained in the system are accessed through the root folder. The root folder is also used for accessing the RAM disc.

ROM: The abbreviation for read-only memory. This memory is not lost when the computer is turned off and is used for storing the operating system, device files, the PAM program, and other system information.

S

scaling points: The points identified as P1 and P2 with a plotter device. The point P1 represents the lower left corner of the plotting area and P2 represents the upper right corner. The locations of these points can be changed with HP-GL commands. When the scaling points are changed, the scale of drawings in the plotter window changes proportionately.

script file: A data file that contains commands and has execute permission. When the name of a script file is entered as a command, the commands in it are executed as if they were entered individually from the command line. Script files are useful for performing frequently needed operations and for executing commands from programs. Script files are classified as a special type of program file.

search path: A list of folders that are searched in order when the name of a file is entered in the command line. The search path is specified by the environment file variable PATH. Certain application programs also use this variable for determining what folders to search in order to locate files not found in the current folder. The search path is not used to locate files entered as parameters.

sequential processing: The serial execution of programs and other processes in the computer. Sequential processing can be specified for individual commands by placing a semicolon after them in the command line. This causes the commands to be executed one at a time. A command following a command with a semicolon is not started until the command preceding it is completed.

shell: A program that is run continuously to provide a user interface to a computer. The PAM program is used as a shell program on the Integral Personal Computer. This interface allows programs to be submitted to the operating system for execution, displaying of information in folders, and provides many other capabilities with shell commands. The shell commands for the PAM program are the built-in commands that can be entered in the command line.

simple file name: A file name that identifies a file by its name and does not include a path name. The names of files displayed in the PAM window are an example of simple file names. Simple file names are used for specifying files in the open folder. *See also* **file name**.

source: In a data transfer operation, the file, folder, or disc from which data is transferred. In a copy operation, the file being copied from is referred to as the source. When a compiler is used, the data file with the statements being compiled is called the source file. *See also* **destination**.

string: A sequence of characters and other symbols that can be used as one unit by a program, command, or other function. A string that only contains characters is referred to as a character string.

subfolder: A folder (directory) that is contained in another folder. All folders stored in the current folder are called subfolders, for example. All folders in a file system are subfolders relative to its root folder.

T

tab stop: A column to which the cursor can be moved with the Tab key.

target: The destination file or folder for an operation that transfers data from one location to another. A parameter is usually used for identifying the target. In a copy operation, for example, the target folder is the folder the source file is copied to.

task: An operation that can be performed by a user, usually as part of a sequence of operations. Complicated tasks can often be broken down into several subtasks that are easier to perform.

terminal device: A logical device associated with each terminal window. Terminal devices are represented by a device file that can be accessed programmatically by the system. Operations with a terminal device involve the outputting of character data to a terminal window and inputting of character data from the keyboard.

text: Alphabetic character data used to represent written material in a data file or window. Text is usually intended for a user to read and not as a specific instruction for the computer.

text editor: An application program that allows you to create data files, enter text, and perform various editing functions such as searching and replacing characters, moving blocks of text, and justifying margins. Some text editors allow you to designate different print modes such as underline or bold print for use with a printer.

U

unit number: A number used to identify a specific disc drive in a disc drive unit. A dual disc drive normally uses 0 for the first drive and 1 for the second drive. The disc unit number is usually beside the slot where discs are inserted. Single disc drives and the internal disc drive have a unit number of 0.

utility: A program designed for use with HP-UX systems that performs a specific function. Utility programs are used for initializing discs, listing folders, and performing many other necessary operations.

V

variable assignment: A sequence of characters that causes a variable to be set to a value. The variable name is usually followed by a specific operator, such as the equal sign, and then by the value it is to represent.

volume name: A one-to-six character name used by single-user systems with removable file systems to identify a particular disc. A volume name is assigned to the disc when it is formatted. When the disc is mounted by the operating system, the volume name is entered into the root directory as the name of the top-level directory on the disc.

volume number: A number used to identify each file system or volume stored on a disc. The only disc devices that are normally used for storing multiple volumes are hard discs. The volume number is incorporated into the device file names for peripheral disc drives so that different volumes on the same hard disc are uniquely identified. The volume number of a file system stored on a flexible disc is 0.

W

waiting list: *See* **printer waiting list**.

window: The part of the display used for a particular program or an alpha or graphics device. Your input to a program and its displayed output are contained in the program's window. When more than one of these programs or devices exists, their windows are stacked on the screen like pieces of paper on a desk. *See also* **active window**.

window banner: The tab on the upper left corner of each window that contains the name of the window.

window manager program: A program responsible for maintaining separate windows on the screen and implementing other window operations. These operations include moving windows and stretching windows. The window manager program can also be used for controlling windows programmatically.

write protect: A method of protecting information stored on a disc. The location of a tab or notch on the disc prevents write operations on the disc. A disc that is write protected can still be used for read operations.

B Keyboard Maps

This appendix correlates specific keys on the keyboard with control sequences and extended characters. The map of keyboard control sequences indicates what control code or escape sequence is returned when the key is pressed. The characters in the map of extended character indicate what characters are produced when a character key is pressed with (Extend char).

Control Sequences

The control codes and escape sequences in the keyboard control sequence map are for the normal mode of an alpha graphics window. HP escape sequences are returned in HP mode and ANSI escape sequences are returned in ANSI mode. The escape sequence to select ANSI mode is `&k1\`. To select HP mode, the escape sequence is `&k0\`. To determine the decimal equivalents for the control codes, refer to the U.S. ASCII character set in appendix C.

Keystroke	Control Code	HP Escape Sequence	ANSI Escape Sequence
Back space	Back space (BS)		
Clear display		&J	&[J
Clear line		&K	&[K
CTRL G	Bell		
CTRL H	Back space (BS)		
CTRL I	Horizontal tab (HT)		
CTRL J	Line feed (LF)		
CTRL M	Carriage Return (CR)		
CTRL N	Shift out (SO)		
CTRL O	Shift in (SI)		
CTRL Q	Device control 1 (D1)		
CTRL S	Device control 3 (D3)		
Delete char		&P	&[P
Delete line		&M	&[M
Enter		&d	&[d
Insert char (if off)		&Q	&[4h
Insert char (if on)		&R	&[4l
Insert line		&L	&[L
Next		&U	&[U
Prev		&V	&[V
Return	Carriage return (CR)		
Shift ➤		&F	&[F

Keystroke	Control Code	HP Escape Sequence	ANSI Escape Sequence
Shift Tab		Ei	E[Z
Shift ▲		ES	E[S
Shift ▲ ◄		E&r1u1L	E[S and E[@
Shift ▲ ►		E&r1u1R	E[S and E[A
Shift ◄		E&r1L	E[@
Shift ▼		ET	E[T
Shift ▼ ◄		E&r1d1L	E[T and E[A
Shift ▼ ►		E&r1d1R	E[T and E[A
Shift ►		E&r1R	E[A
Shift Clear line		EG EK	E[2K
Shift Clear display		Eh EG	E[2J
Tab	Horizontal tab (HT)		
►		Eh	E[H
▲		EA	E[A
▲ ◄		EA and ED	E[A and E[D
▲ ►		EA and EC	E[A and E[C
◄		ED	E[D
▼		EB	E[B
▼ ◄		EB and ED	E[B and E[D
▼ ►		EB and EC	E[B and E[C
►		EC	E[C

(Extend char) is enabled only when the active window is an alpha window in normal mode. Only the keys that generate extended characters are included. If no shifted character is indicated, then the same character is produced whether or not (Shift) is pressed. The system accesses the extended characters from the Roman Extension character set.

Many foreign characters in the character set can be produced by designating an extended character and then entering a normal character. To designate the extended character, press (Extend char) with one of the following keys: (I), (R), (T), (U), or (Y). This produces a character that will appear above the next character entered. To produce á, for example, press (Extend char)(R) followed by (A). If a combination does not produce an extended character then the normal character is displayed.

Extended Character Keyboard Map

Key	Extended Character	Shifted Extended Character
,	<	
-	_	
.	>	
/	_	
0	¿	
1	¡	
2	@	
3	#	
4	‡	
5	½	
6	^	
7	\	
8	[	{
9	]	}
;	£	
=	±	
A	ä	Ä
B	■	
C	ç	Ç
D	ð	Ð
E	æ	Æ
F	ƒ	

Key	Extended Character	Shifted Extended Character
G	ü	
H	¥	
I	~	~
J	‡	
K	¢	
L	£	
M	º	
N	ª	
O	ø	Ø
P	þ	Þ
R	´	´
S	ß	
T	`	`
U	¨	¨
V	§	
W	~	
X	¤	¤
Y	^	^
[	ª	
]	¦	
`	«	»

C Character Set

The Roman8 character set consists of the standard U.S. ASCII and the Roman Extension character sets. Each element of a character set is assigned a number (in the range 0 through 255) that the system uses for identifying characters and control codes. Codes and characters are normally produced from the keyboard by pressing certain keys.

The first half of the character set, decimal 0 through decimal 127, is identical to the standard character set used on many other computer systems. The second half, decimal 128 through 255, contains foreign characters and other special characters.

US ASCII Character Set

ASCII Char.	Equivalent Forms			
	Dec	Binary	Oct	Hex
NUL	0	00000000	000	00
SOH	1	00000001	001	01
STX	2	00000010	002	02
ETX	3	00000011	003	03
EOT	4	00000100	004	04
ENQ	5	00000101	005	05
ACK	6	00000110	006	06
BEL	7	00000111	007	07
BS	8	00001000	010	08
HT	9	00001001	011	09
LF	10	00001010	012	0A
VT	11	00001011	013	0B
FF	12	00001100	014	0C
CR	13	00001101	015	0D
SO	14	00001110	016	0E
SI	15	00001111	017	0F
DLE	16	00010000	020	10
DC1	17	00010001	021	11
DC2	18	00010010	022	12
DC3	19	00010011	023	13
DC4	20	00010100	024	14
NAK	21	00010101	025	15

ASCII Char.	Equivalent Forms			
	Dec	Binary	Oct	Hex
SYNC	22	00010110	026	16
ETB	23	00010111	027	17
CAN	24	00011000	030	18
EM	25	00011001	031	19
SUB	26	00011010	032	1A
ESC	27	00011011	033	1B
FS	28	00011100	034	1C
GS	29	00011101	035	1D
RS	30	00011110	036	1E
US	31	00011111	037	1F
space	32	00100000	040	20
!	33	00100001	041	21
"	34	00100010	042	22
#	35	00100011	043	23
$	36	00100100	044	24
%	37	00100101	045	25
&	38	00100110	046	26
'	39	00100111	047	27
(	40	00101000	050	28
)	41	00101001	051	29
*	42	00101010	052	2A
+	43	00101011	053	2B

US ASCII Character Set (continued)

ASCII Char.	Equivalent Forms			
	Dec	Binary	Oct	Hex
,	44	00101100	054	2C
-	45	00101101	055	2D
.	46	00101110	056	2E
/	47	00101111	057	2F
0	48	00110000	060	30
1	49	00110001	061	31
2	50	00110010	062	32
3	51	00110011	063	33
4	52	00110100	064	34
5	53	00110101	065	35
6	54	00110110	066	36
7	55	00110111	067	37
8	56	00111000	070	38
9	57	00111001	071	39
:	58	00111010	072	3A
;	59	00111011	073	3B
<	60	00111100	074	3C
=	61	00111101	075	3D
>	62	00111110	076	3E
?	63	00111111	077	3F
@	64	01000000	100	40

ASCII Char.	Equivalent Forms			
	Dec	Binary	Oct	Hex
A	65	01000001	101	41
B	66	01000010	102	42
C	67	01000011	103	43
D	68	01000100	104	44
E	69	01000101	105	45
F	70	01000110	106	46
G	71	01000111	107	47
H	72	01001000	110	48
I	73	01001001	111	49
J	74	01001010	112	4A
K	75	01001011	113	4B
L	76	01001100	114	4C
H	77	01001101	115	4D
N	78	01001110	116	4E
O	79	01001111	117	4F
P	80	01010000	120	50
Q	81	01010001	121	51
R	82	01010010	122	52
S	83	01010011	123	53
T	84	01010100	124	54
U	85	01010100	125	55

US ASCII Character Set (continued)

ASCII Char.	Equivalent Forms			
	Dec	Binary	Oct	Hex
V	86	01010110	126	56
W	87	01010111	127	57
X	88	01011000	130	58
Y	89	01011001	131	59
Z	90	01011010	132	5A
[	91	01011011	133	5B
\	92	01011100	134	5C
]	93	01011101	135	5D
^	94	01011110	136	5E
_	95	01011111	137	5F
'	96	01100000	140	60
a	97	01100001	141	61
b	98	01100010	142	62
c	99	01100011	143	63
d	100	01100100	144	64
e	101	01100101	145	65
f	102	01100110	146	66
g	103	01100111	147	67
h	104	01101000	150	68
i	105	01101001	151	69
j	106	01101010	152	6A

ASCII Char.	Equivalent Forms			
	Dec	Binary	Oct	Hex
k	107	01101011	153	6B
l	108	01101100	154	6C
m	109	01101101	155	6D
n	110	01101110	156	6E
o	111	01101111	157	6F
p	112	01110000	160	70
q	113	01110001	161	71
r	114	01110010	162	72
s	115	01110011	163	73
t	116	01110100	164	74
u	117	01110101	165	75
v	118	01110110	166	76
w	119	01110111	167	77
x	120	01111000	170	78
y	121	01111001	171	79
z	122	01111010	172	7A
{	123	01111011	173	7B
|	124	01111100	174	7C
}	125	01111101	175	7D
~	126	01111110	176	7E
DEL	127	01111111	177	7F

Roman Extension Character Set

ASCII Char	Equivalent Forms	
	Dec	Binary
	128	10000000
	129	10000001
	130	10000010
	131	10000011
	132	10000100
	133	10000101
	134	10000110
	135	10000111
	136	10001000
	137	10001001
	138	10001010
	139	10001011
	140	10001100
	141	10001101
	142	10001110
	143	10001111
	144	10010000
	145	10010001
	146	10010010
	147	10010011
	148	10010100
	149	10010101

ASCII Char	Equivalent Forms	
	Dec	Binary
	150	10010110
	151	10010111
	152	10110000
	153	10011001
	154	10011010
	155	10011011
	156	10011100
	157	10011101
	158	10011110
	159	10011111
space	160	10100000
À	161	10100001
Å	162	10100010
È	163	10100011
Ê	164	10100100
Ë	165	10100101
Î	166	10100110
Ï	167	10100111
´	168	10101000
`	169	10101001
^	170	10101010
¨	171	10101011

Roman Extension Character Set (continued)

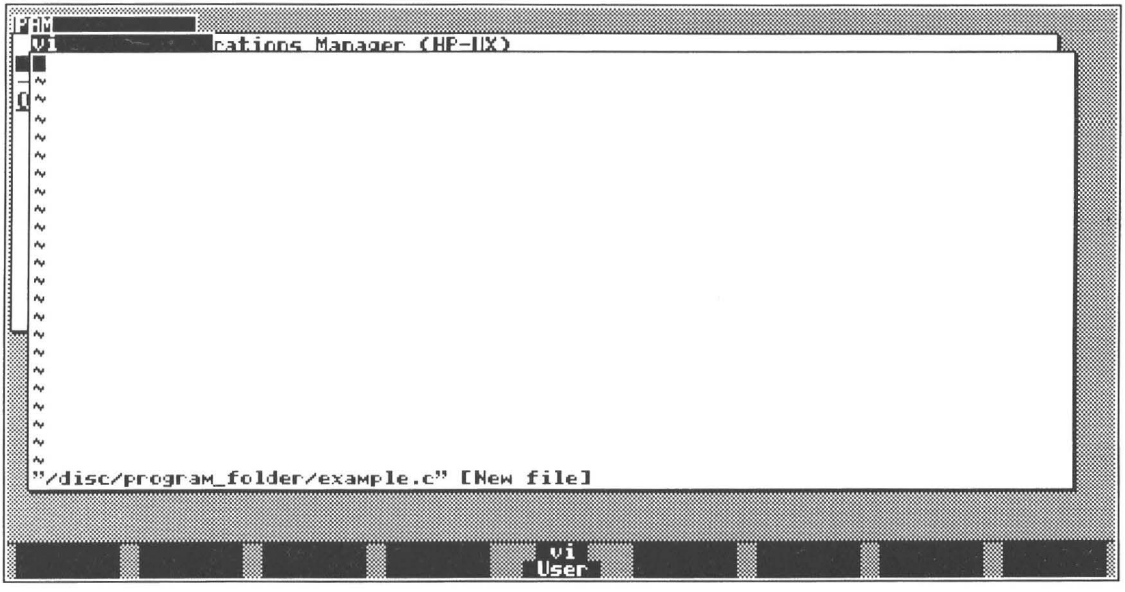
ASCII Char	Equivalent Forms	
	Dec	Binary
~	172	10101100
Ù	173	10101101
Ú	174	10101110
£	175	10101111
¯	176	10110000
	177	10110001
	178	10110010
¤	179	10110011
Ç	180	10110100
ç	181	10110101
Ñ	182	10110110
ñ	183	10110111
¡	184	10111000
¿	185	10111001
Ø	186	10111010
£	187	10111011
¥	188	10111100
§	189	10111101
ƒ	190	10111110
¢	191	10111111
à	192	11000000

ASCII Char	Equivalent Forms	
	Dec	Binary
è	193	11000001
ò	194	11000010
ù	195	11000011
á	196	11000100
é	197	11000101
ó	198	11000110
ú	199	11000111
à	200	11001000
è	201	11001001
ò	202	11001010
ù	203	11001011
ä	204	11001100
ë	205	11001101
ö	206	11001110
ü	207	11001111
Ä	208	11010000
î	209	11010001
Ø	210	11010010
Æ	211	11010011
à	212	11010100
î	213	11010101

Roman Extension Character Set (continued)

ASCII Char	Equivalent Forms	
	Dec	Binary
ø	214	11010110
æ	215	11010111
Ä	216	11011000
ì	217	11011001
Ö	218	11011010
Ü	219	11011011
É	220	11011100
ï	221	11011101
ß	222	11011110
Ô	223	11011111
Á	224	11100000
Ã	225	11100001
ã	226	11100010
Ð	227	11100011
ð	228	11100100
Í	229	11100101
Ì	230	11100110
Ó	231	11100111
Ò	232	11101000
Õ	233	11101001
õ	234	11101010

ASCII Char	Equivalent Forms	
	Dec	Binary
Š	235	11101011
š	236	11101100
Ú	237	11101101
Ÿ	238	11101110
ÿ	239	11101111
Þ	240	11110000
þ	241	11110001
	242	11110010
	243	11110011
	244	11110100
	245	11110101
–	246	11110110
‡	247	11110111
±	248	11111000
ª	249	11111001
º	250	11111010
«	251	11111011
■	252	11111100
»	253	11111101
±	254	11111110
	255	11111111



The *vi* editor has created a buffer area in memory and expects to create a new file called **/disc/program_folder/example.c** at the end of the editing session.

Each empty line below the cursor is marked with a *tilde*. The *tilde* "~" is a special character used to mark the end of the text in a file that already exists, or in the case of a new file, to show that the file is empty. The tildes will not become part of the text in your file.

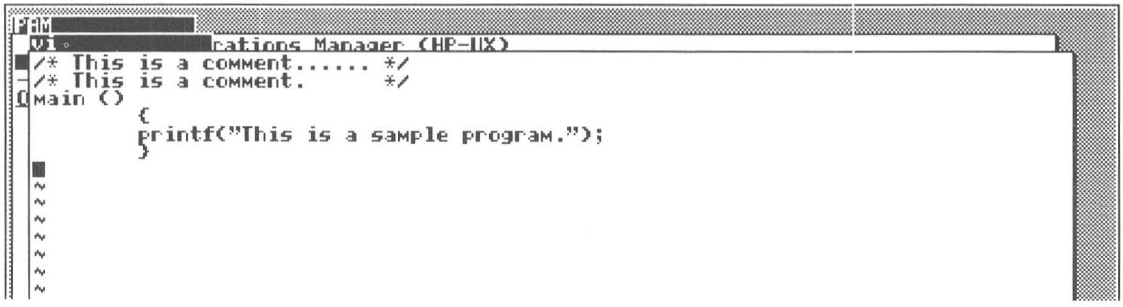
Entering Your Program

Before we save this file, let's put something in it. When the *vi* editor is first invoked, it is in the command mode. It is now waiting for your command. For this example, let's type in a program. To enter the append mode, type in **a**.

The editor is now in the append mode and waiting for you to type in text.

Like most commands used by the *vi* editor, the **a** command will not be echoed on the screen. Even though the command does not print on the screen, the editor will move the cursor and then wait for you to enter text.

Type in the following program, ignoring any mistakes you may make. (You will learn how to correct errors later in this section.)



As you type in the text, note that the tilde (~) on each line disappears as you begin typing on that line. Also, if the line is longer than 80 characters, the line of text will wrap around to the next line on the screen. (A "line" to the *vi* editor is a string of up to 512 characters terminated by a (Return). If you type in more than 512 characters without pressing the (Return) key, the editor will respond with an error message.)

To terminate the input mode and return to the command mode, press (ESC).

You are now back in the command mode. Because the editor will not prompt you or otherwise let you know what mode you are currently in, you may want to be sure you really are back in the command mode by pressing the (ESC) key several times. After the first (ESC), the *vi* editor will beep each time you press the (ESC) key to tell you that it expects a command.

Saving Your Text File

To save the text in the file **/disc/program_folder/example.c** and stop the *vi* program, first insert the **/disc** disc. Next type in **ZZ** (*uppercase*).

You do not need to hit the (Return) key with the **ZZ** command. The *vi* editor will respond by displaying the following line at the bottom of the screen:

```
"/disc/program_folder/example.c" [New file]
7 lines, 114 characters
```

You have now terminated the *vi* program and saved your file. The *vi* window is now "dead" and will disappear when the next PAM command is executed. To return to PAM, press (Shift)(Select) or (Stop).

This will shuffle or destroy the *vi* window.

Using the Cursor Commands

This section describes a few commands which will enable you to move around in your file. Try each of the commands as they are discussed to familiarize yourself with them.

To return to our sample program we must again invoke the *vi* editor. Let's assume you do not have a peripheral disc drive available. This will allow us to introduce the **:e** command.

First insert the applications disc; then type in `vi` (Return).

The *vi* editor will start and create a new file area in the buffer memory. To edit our existing program, use the **:e** (edit) command. First insert the **/disc** disc, then type in:

```
:e /disc/program_folder/example.c (Return).
```

The existing file will be copied into the *vi* buffer and the cursor will be positioned at the start of the file in the *vi* window.

After the *vi* window appears, you may use the following basic cursor movement commands to move around your file while in the *command mode* (once in the input mode, only a few commands or keys such as the space, tab, and backspace keys may be used to position the cursor).

Basic Cursor Movement Commands

Command	Action
h, (►)	Moves the cursor one space to the left. The (Backspace) key also moves the cursor one space to the left.
j, (▼)	Moves the cursor down one line in the same column.
k, (▲)	Moves the cursor up one line in the same column.
l, (►)	Moves the cursor one space to the right. The SPACE bar also moves the cursor one space to the right.
+	Move to start of next line.
-	Move to start of previous line.
$	Move to end of line.
(►)	Move cursor to home position on screen.
0	Move to start of line.

(For other cursor movement keys, refer to *The Vi Editor*.)

Adding, Deleting and Correcting Text

The *vi* editor provides several simple commands to add, delete or change text in your file. This section introduces a few of the most frequently used commands. (For a complete description of all the *vi* commands, refer to *The Vi Editor*.)

Commands for Adding, Deleting and Correcting Text

Command	Action
i	Insert
a	Append
o, **O**	Insert line
x	Delete character
dd	Delete line
r	Replace
u	Undo
Y	Yank
P	Put

The *vi* editor will only accept a specific combination of letters and will always distinquish between uppercase and lowercase letters. For example, **o** and **O** are *not* the same command, and, unlike some other editors you may have used, it will not accept the name *insert* for the **i** command.

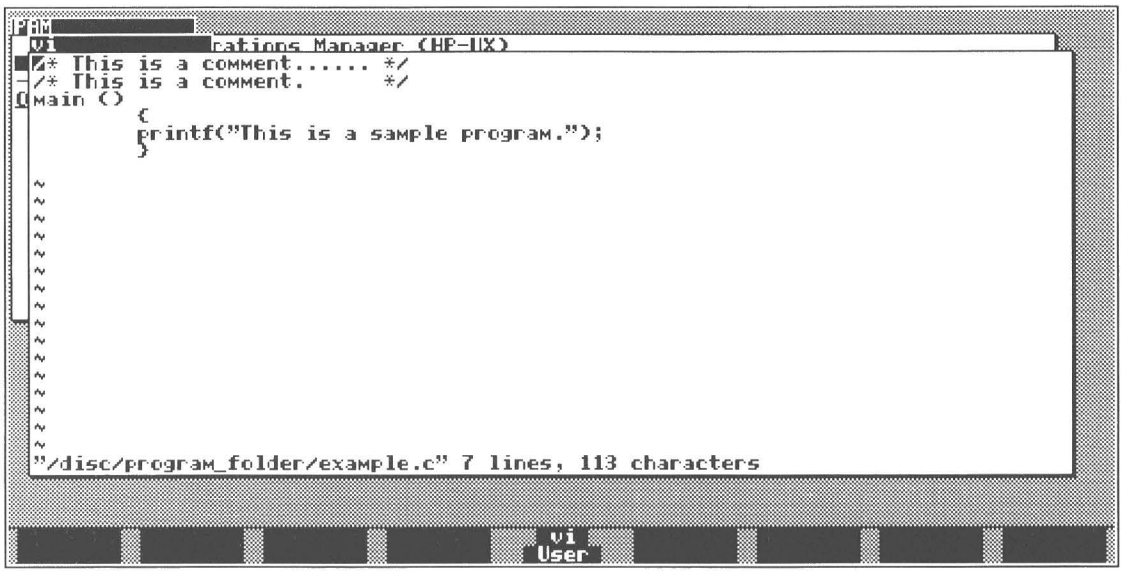
Remember that all commands must be entered from the command mode. To make sure you are in the command mode, press the (ESC) key.

Appending and Inserting Text

The a and i Commands. The **i** and **a** commands are used for insertion and appending text, respectively. The **i** command places text to the left of the cursor, and the **a** command places text to the right of the cursor.

You may insert or append both characters and lines of text with the **i** and **a** commands. To type in more than one line of text, press (Return) at the place in your text where you want the new line.

You should now be in the file **/disc/program_folder/example.c**, and ready to insert new text. We will use this program as an example.



Since you are already familiar with the append command, let's try using the **i** command. For example, to insert the word **too** in the second comment line, first position the cursor after the word **This** in the second line using the cursor movement commands. Next type in **i** to get into the insert mode.

Now type in (Space) too (ESC).

Your first two lines of the program should read like this:

```
/* This is a comment..... */
/* This too is a comment.       */
```

The o and O Commands. The **o** command creates a new line after the line containing the cursor. The **O** command creates a new line before the line containing the cursor. Both commands leave you in the text input mode. Pressing (Return) after an **o** or **O** command creates an additional new line.

Like the **a** and **i** commands, the **o** and **O** commands are invoked only from the command mode (remember to press the (ESC) key to get into the command mode or to clear any partially formed command).

To use the **o** command in the example above, let's add a line between the second and third lines. First check to be sure the cursor is positioned on the second line, then type in **o**.

The bottom three lines should move down creating an empty line between the second and third lines of text.

```
/* This is a comment..... */
/* This too is a comment.       */

main ()
```

To leave this mode, press (ESC).

To use the **O** command to create a new line between the first and second lines of text, press (k) to move the cursor to the second line, then type in **O**.

The second line of text and all the lines below it will move down one line, creating an empty line between the first and second lines of text, leaving you in the append mode.

```
/* This is a comment..... */

/* This too is a comment.       */

main ()
```

Again, to leave this mode, press (ESC).

Deleting and Replacing Text

The x Command. The **x** command deletes the character marked by the cursor. You can delete more than one character at a time by preceding **x** with a number. For example, **5x** deletes five characters beginning at the cursor.

To delete the five extra periods at the end of the first line of text, position the cursor on the second period, then type in **5x**.

The period under the cursor and the four periods to the right of the cursor will be deleted.

```
/* This is a comment. */

/* This too is a comment.      */

main ()
```

The dd Command. To delete a line, place the cursor on the line you want to delete and use the **dd** command. For example, to delete the first blank line in the example above, move the cursor down one line to position it on the first blank line, then type in **dd**.

The first blank line will be deleted and the lines below it will move up.

```
/* This is a comment. */
/* This too is a comment.      */

main ()
```

The r Command. The **r** command is used to replace an incorrect letter or change two letters around. In our example, let's replace **m** in the word **main** with **M**, position the cursor over the **m**, then type in **r**. The editor is now waiting for the replacement character. Type in **M**.

The letter **m** will be replaced with the letter **M**, and the cursor will be positioned over the **M**.

```
/* This is a comment. */
/* This too is a comment.       */

Main ()
```

Undoing Commands

If for some reason you decided you didn't want to make a particular change after all, you can correct your mistake by using the undo command **u**. When you execute the **u** command, you undo the most recent change in your text.

To use the above example and change the **M** back to the lowercase **m**, (if you are still in the command mode) type in **u**.

This will undo the changes caused by the most recently executed command (in this case the **r** command) that changed the text. Note that if you have executed any other command since the **r** command, the **u** command will undo *that* command.

```
/* This is a comment. */
/* This too is a comment.       */

main ()
```

(Note that the cursor is positioned where it was before the **r** command was executed.)

Moving Lines of Text

The Y Command. The yank command (**Y**) allows you to place text in a buffer temporarily so it can be moved to another location. The **Y** command can be prefixed by a number to yank that many lines from the file and put them into a buffer.

The P Command. The put command **P** (uppercase) takes the line of text out of the buffer, and puts it before the cursor.

To copy the first line and put it after the second line in our example, first position the cursor at the beginning of the first line; next type in **Y**.

The first line will remain on the screen, and its contents will be placed into an unnamed buffer. Next, position the cursor at the beginning of the empty line after the second line, type in **P**.

This puts a copy of the first comment between the second comment line and the empty line following it.

```
/* This is a comment. */
/* This t/* This too is a comment.      */
/* This is a comment. */

main ()
```

F

Writing Text to a Disc File

The **:w** (colon **w**) command is used to write the current contents of the *vi* buffer onto a file. The contents of the *vi* buffer remain unchanged.

If you specified a file name when the *vi* editor was invoked, the **:w** command will use the name of the file you are editing or creating, and writes to that file by default.

To write to the file **/disc/program_folder/example.c** on the **/disc** disc, first insert the disc in the internal disc drive.

Next press (ESC). This gets you out of any *vi* command mode you might be in. Then type in **:w** (Return).

The command will be echoed at the bottom of the *vi* screen. When the command is finished, the cursor will return to the text area. A copy of the *vi* buffer has now been written on the disc.

If you have not specified a file name up to this point, you must supply a name with the **:w** command.

After executing the **:w** command you may continue editing your file or use the **:e** command to edit another file. If you want to exit the *vi* editor and save your file, use the **ZZ** command.

The *vi* editor will save your editing changes in the file name you previously specified, write the character and line counts at the bottom of the screen, and stop execution.

After exiting *vi* using the **ZZ** command, the *vi* window is "dead," which means the *vi* program has been terminated and the window will disappear when the next PAM command is executed. To get back into PAM, press (Shift)(Select) to shuffle the windows.

If you just want to quit and you don't want to save your editing changes, type in **:q!** (Return).

Again, to return to the PAM window, shuffle the windows.

Where To Find More Information on Vi

For more in-depth information on the *vi* editor, refer to the article "The Vi Editor" in *HP-UX Concepts and Tutorials*, Vol. 1, Text Processing and Formatting.

To quickly find a command, refer to the *vi* quick reference guide.

For information about the syntax used in invoking *vi*, refer to the on-line manual pages from the *HP-UX Reference*.

Index

A

Absolute file names, related to path names, A-1
Absolute path names, 3-9, A-1
Accessing files, 1-11, 3-22
Action, setting, 3-46
Active plotter, 5-11, 5-20
Active printer, 5-11, 5-17
Active window
  changing size, 2-14 thru 2-15
  connected to keyboard, 1-6
  connected to mouse, 1-6
  controlling, 2-11 thru 17
  defined, A-1
  menus affect, 2-1 thru 2-3
  moving, 2-13 thru 2-14
  pausing output, 2-16
  scrolling, 2-9
  selecting with mouse, 2-4
  stretching, 2-14 thru 2-15
  suspending program, 2-16
Adding discs, *mount_disc* program, 4-55 thru 4-56
Addresses
  defined, A-6
  disc drive, 5-15
  HP-IB, 5-17
  peripheral, 5-4
  plotter, 5-4
  preset, 5-4
  printer, 5-4
Adventure game program, 1-2
Alpha windows
  controlling, 2-19 thru 2-21
  defined, A-1
  memory requirement, 1-13
  menu, 2-20 thru 2-21
  related to printer control, D-1
Anchor, 2-13, A-1
ANSI escape sequences, B-1 thru B-3
Applications
  can redefine keys, 2-8
  controlled by PAM, 1-7
  define function keys, 2-8
  multiple, 1-5 thru 1-6
  use data files, 1-9
Arrow keys, defined, A-2
ASCII character set, listing, C-2 thru C-4
Autost files, 3-37, 3-40, 4-7 thru 4-8, 5-10, A-2
Autostarting the computer, 4-7 thru 4-8

B

Banners, defined, A-21
BASIC development package, 1-2
beep_off program (Utilities disc), 4-9
beep_on program (Utilities disc), 4-10
Beeper, 4-9, 4-10
Binary numbers, defined, A-2
Block devices, defined, A-2
Bold printing, D-11
Break, sending to host computer, 2-10
Buffers, defined, A-2
Built-in devices, defined, A-2
Bus expander, 5-8
Bytes, defined, A-3

C

C development package, 1-2
Cable, serial interface, 5-3
Calculations, performing, 4-74 thru 4-75
Calculator program (Standard Applications disc), 4-74 thru 4-75
cd command (PAM), 3-8, 3-43, 4-11 thru 4-12
Character devices, defined A-3
Character fonts, defined, A-3
Character sets, A-3, C-1 thru C-7
  creating, 4-35 thru 4-36
  downloading into windows, 4-82 thru 4-83
Characters
  deleting, 2-9
  inserting, 2-9
  special, B-4 thru B-5
chmod program (HP-UX disc), 1-11, 3-38, 4-14 thru 4-17
Clock, checking and setting, 4-88 thru 4-89
close command (PAM), 4-13
Closing folders, 3-10, 4-13
Command area (PAM), 1-8
Command line, 3-36, A-3
Commands, defined, A-3
Compilers, defined, A-4
Compiling a data file, 3-10
Concurrent execution, defined, A-4
Connecting programs, 3-31 thru 3-33
Control characters. *See* Control codes
Control codes
  keyboard map, B-1 thru B-3
  defined, A-4
  display functions mode, A-7
  typing, 2-6, 2-21
  used in alpha window, A-1
  used with printer, D-1 thru D-3, D-7, D-21
Control sequences, defined, A-4
copy command (PAM), 3-3, 3-7, 3-18, 3-21, 4-18 thru 4-20
copy_disc program (Utilities disc), 4-21 thru 4-25
Copying discs, 4-21 thru 4-25
Copying files, 1-12, 3-18, 4-18 thru 4-20
Creating files, 3-10
Creating folders, 4-53 thru 4-54
Current device file, 5-16
Current loop interface, 5-9
Cursor
  defined, A-4
  in *vi* editor, F-6 thru F-7
  moving, 2-9

D

Data files, 1-9, 1-11, A-5. *See also* Files
Data paths, defined, A-5
Debugging, defined, A-5
delete command (PAM), 3-12, 4-27 thru 4-28
Deleting files and folders, 3-12, 4-27 thru 4-28
Destination, defined, A-6
dev folder, 1-11, 3-13, 5-8, 5-9, 5-11, 5-13 thru 5-15, 5-21, A-6
Device addresses, defined, A-6
Device files. *See also* Files
  access physical devices, 1-11, 5-9, 5-11
  default, 1-12
  defined, A-6
  type of file, 1-9
Device names, defined, A-6
Devices
  accessed using device files, 1-9, 1-11, 5-1, 5-11
  identification, 5-10
  default, defined, A-6
Diacritical marks, typing, 2-7, B-5
Diagnostic disc, 1-2
diagnostic program (Diagnostic disc), 4-29 thru 4-30
Directories, A-7, A-9. *See also* Folders
Disc drive addresses, 5-15
Disc drives, external, 1-1, 4-77 thru 4-79, 5-12, 5-14

Disc files, accessing, 5-14
Disc formatter program, 1-9
Disc performance, enhancing, 4-24
Discs
  adding to system, 4-55 thru 4-56
  copying files between, 1-12
  double-sided, 5-15
  electronic. *See* Electronic disc
  formatting, 4-37 thru 4-41, 5-14, A-9
  free space, 4-84 thru 4-85
  included with computer, 1-2
  initializing, 5-14, A-11
  inserting, 5-13
  mounting, 5-13, A-12
  outer folder of, 1-9
  preparing, 4-37 thru 4-41
  removing, 5-13
  renaming, 4-69 thru 4-72
  single-sided, 5-15
  unmounting, 4-90 thru 4-91
  use, 1-8
  verifying integrity, 4-92 thru 4-95
  volume label, 5-14
disp_time_off program (Utilities disc), 4-31
disp_time_on program (Utilities disc), 4-32
Display, keeping turned on, 4-31
Display bar, defined, A-7
Display functions mode
  enabling, 2-21
  defined, A-7
  used with printer, D-19
Display pointer
  controlled by mouse, 2-4
  defined, A-7
  moving, 2-11
  retrieving hidden window, 2-17
  selects choice, 2-10
  while moving window, 2-13
  while stretching window, 2-14
Display timeout, disabling and enabling, 4-31, 4-32
Displaying files, 4-96 thru 4-97
Drive name, 5-15 thru 5-17
Driver program, 5-1
Drivers
  defined, A-7
  device, 5-3
  disc drive, 5-3
  HP-IB, 5-8
  installing, 5-2, 5-9
  interface, 5-9
  serial interface, 5-8, 5-10

E

echo command (PAM), 3-25, 4-33 thru 4-34
Echoing messages, 4-33 thru 4-34
Editing keys, 2-8 thru 2-9, A-8
Electronic disc, 1-12, 3-21. *See also* RAM disc
.environ file, 3-8, 3-37, 3-40 thru 3-44
Environment, PAM, 4-42 thru 4-44
Environment files, 3-37, 3-40 thru 3-47, A-8
Environments, 3-47, A-4
Erasing files, *delete* command, 4-27 thru 4-28
Error messages, E-1 thru E-7
Errors, clearing in graphics window, 2-22
Escape sequences
  ANSI, B-1 thru B-3
  defined, A-8
  display functions mode, A-7
  graphics, D-18 thru D-19
  HP, B-1 thru B-3
  linking, D-5 thru D-6
  parameterized, D-4 thru D-6
  two-character, D-4
  types, B-1 thru B-3
  typing, 2-6, 2-21
  used in alpha window, A-1
  used with printer, D-1, D-3 thru D-6, D-18 thru D-19, D-21 thru D-22
Execute permission. *See* Permissions
Extended characters, 2-7, A-8
Extension cords, 5-7
External disc drive, 5-14

F

fedit program (Standard Applications disc), 4-35 thru 4-36
File names, 5-10, A-1, A-8
File path names, defined, A-9
File system, 1-8 thru 1-12, A-9
Files. *See also* Data files; Device files; Program files
  accessing, 1-11, 3-22
  allocated in RAM, 1-12
  Autost, 3-37
  copying, 1-12, 3-18, 4-18 thru 4-20
  data type, 1-9, 1-11
  deleting, 3-12
  description, 1-8
  device type, 1-9, 5-11
  environment, 3-37
  execute permission, 1-11
  hidden, 1-8, 3-13
  internal, 5-11
  listing, 4-96 thru 4-97, 4-49 thru 4-51
  lp, 5-11
  modes, A-9
  moving, 3-14 thru 3-16, 4-57 thru 4-59
  names, 1-8
  operations, 3-37
  permissions, 1-11, 3-38, 4-14 thru 4-17, A-9
  placing in root folder, 1-12
  plotter, 5-11
  printing, 3-24, 4-63 thru 4-64
  program type, 1-9, 1-11
  read permission, 1-11
  renaming, 3-17
  script type, 3-37, 5-10
  security, 1-11
  simple names, A-18
  size defined, A-9
  type affects permissions, 1-11
  types, 1-9, A-9
  viewing, 3-23, 4-96 thru 4-97
  write permission, 1-11
Folder, **dev**, 5-8, 5-9, 5-11, 5-13, 5-15 thru 5-16, 5-21
Folder area (PAM), 1-8, 1-11
Folders
  contain files, 1-8
  creating, 1-12, 4-53 thru 4-54
  defined, A-9
  deleting, 3-12
  displaying information, 1-11
  group files, 1-9 thru 1-10
  hidden, 3-13
  home, A-10
  listing, 4-49 thru 4-51
  open, 1-11
  parent defined, A-13
  rereading, 4-73
  root, 3-11, 5-14, 5-17. *See also* Root folder
  subfolders defined, A-18
  type of file, 1-9
Font editor, 4-35 thru 4-36
Fonts, 4-82 thru 4-83, A-3
fonts folder (Standarad Applications disc), 4-82 thru 4-83
format_disc program (Utilities disc), 4-37 thru 4-41, 5-14
Formatting discs, 4-37 thru 4-41, A-9
Function keys, 2-1, 2-8

G

getenv command (PAM), 3-43, 4-42 thru 4-44
Graphic design program, 1-2
Graphics
  drawing, 2-21 thru 2-24
  escape sequences for printing, D-18 thru D-19
  printing, 2-21, D-18 thru D-19
  setting printer dot density, D-18

Graphics windows
clearing, 2-22, 2-24
clearing errors, 2-22
creating, 2-23, 5-21
defined, A-10
memory requirement, 1-13
menu, 2-21 thru 2-22
removing, 2-24
scrolling, 2-23
showing, 4-2
size, 2-21

H

Hard disc, 5-2
Hardware, checking, 4-29 thru 4-30
-help option, for online help, 4-6
Hidden files, 1-8, 3-13, A-10
Hidden folders, 3-13
Hidden windows, 2-16, 2-17, A-10
Home folders, defined, A-10
HOME variable (PAM), 4-42
Host computer, sending break to, 2-10
HP escape sequences, B-1 thru B-3
HP-GL, loading the command set, 4-45 thru 4-46
HP-IB
accessories, 5-4
adapter, 5-4
address, 5-2, 5-10, 5-15, 5-17
cable, 5-1, 5-2, 5-4, 5-5
defined, A-10
interface, 5-2
metric conversion kit, 5-5
peripherals, 1-1, 5-4, 5-6
plotter, 5-21
HP-UX
defined, A-10
commands disc, 1-2, 4-1
documentation (Utilities disc), 4-6
operating system, 1-1, 1-3

I

Identifiers, 4-3
IEEE-488-1978, 5-4
Initialization, 4-37 thru 4-41, A-11
Inserting a disc, 5-13
Interactive programs (utilities), 4-5
Interfaces. *See also* HP-IB
capabilities, 1-3
connect to peripherals, 5-2
current loop, 5-9
driver, 5-9
serial, 5-9
internal file, 5-11
Internal memory, 1-1, 1-8. *See also* RAM
Internal printer. *See* Printer, internal
I/O expanders, 1-1
I/O ports, 5-2, 5-4

K

Key labels. *See* Menus
Keyboard
connected to active window, 1-6
diagram, 2-5
key groups, 2-5
map, B-1 thru B-5
plotting from, 2-23 thru 2-24
tied to active window, 2-11
use, 2-5 thru 2-11

L

Line spacing, setting for printer, D-12
Lines (graphics), drawing, 2-23
Lines (text)
changing in *vi* editor, F-8 thru F-14
deleting and inserting, 2-9
Listing files and folders, 4-49 thru 4-51
load_hpgl program (System disc), 4-45 thru 4-46
load_serial program (System disc), 4-47 thru 4-48
Lowercase letters
in escape sequences, D-3
in file names, 1-8, 4-3
typing, 2-5, 2-6
lp device file, 4-63, 4-65 thru 4-66, 5-11, 5-18, D-1
ls program (HP-UX disc), 3-13, 4-49 thru 4-51

M

makefolder command (PAM), 3-11, 3-21, 4-53 thru 4-54
Making new folders, 4-53 thru 4-54
Margins, left and right, 2-20 thru 2-21
Memory, internal, 1-1, 1-8, 4-84 thru 4-85. *See also* RAM
Memory, requirements for alpha and graphics windows, 1-13
Memory modules, 1-13
Menus. *See also* System menu; User menu
  choosing items, 2-4
  define function keys, 2-8
  defined, A-11
  labels, A-11
  operation, 2-1 thru 2-3
  toggling, 2-8, 2-10
  turning off and on, 2-10
Messages, E-1 thru E-7
Modems, 1-1, 5-9
Modes, file permissions, A-9
Mounted discs, 5-13, A-12
mount_disc program (Utilities disc), 4-55 thru 4-56, 5-13 thru 5-14
Mouse
  button functions, 2-4
  connected to active window, 1-6
  connection, 2-4
  defined, A-12
  moving window, 2-13
  selects active window, 2-11
  selects menu items, 2-4
move command (PAM), 3-21, 4-57 thru 4-59
Moving files, 3-14 thru 3-16, 4-57 thru 4-59
Multitasking, 1-5 thru 1-6

N

Non-sequential processing, 3-35
Null string, defined, A-12

O

Octal numbers, defined, A-12
Online documentation, 4-6
Open folder, 1-1, 3-3, A-12
Opening folders, 4-11 thru 4-12
Operating systems, defined, A-13
Overstrike printing, D-15
Overwriting, defined, A-13

P

Page length, setting for printer, D-14
PAM
  accessing files, 3-22
  active environment, 3-8
  built-in program, 1-1
  changing folders, 3-8
  closing folders, 3-10
  command area, 1-8, 3-2
  command line, 3-6
  command memory, 3-7
  commands, 4-1 thru 4-97
  copying files, 3-3, 3-7, 3-18
  creating files in, 3-10
  creating folders, 3-11
  defined, A-13
  deleting files, 3-12
  deleting folders, 3-12
  file manipulation, 3-8
  folder area, 1-8, 3-3
  function keys, 3-4
  home folder, 3-8
  moving files, 3-14 thru 3-16
  moving the display pointer, 3-9
  opening folders, 3-8
  operation, 1-7 thru 1-8, 3-1 thru 3-47
  printing files, 3-24
  redefines keys, 2-8
  renaming files and folders, 3-17
  rereading folders, 3-3, 3-10

root folder, 3-8
special key summary, 3-5
using programs from, 3-9, 3-10
user Menu, 3-4
uses a window, 1-5
viewing files, 3-23
window, 3-1, A-13
Parameters, command and program, 4-3 thru 4-5
Parent folders, defined, A-13
Path names
absolute, A-1
access files, 1-11
defined, A-14
related to file names, A-8, A-9
relative, 1-11, A-16
PATH variable (PAM), 4-42
Paths, search, defined, A-17
Pen, controlling, 2-22, 2-23
Perforation skip, setting for printer, D12 thru D13
Performance, accessing disc files, 4-24
Peripheral devices
addresses, 5-4
connecting, 5-4
defined, A-14
disc drives, 5-12
installing, 5-1
plotters, 5-19
printers, 5-17
using, 5-1, 5-2, 5-10
Permissions, file security, 1-11, 4-14 thru 4-17, A-9
Personal Applications Manager. *See* PAM
Pipes, 3-31 thru 3-33, A-14
Pitch (printer), A-14, D-9 thru D-12
plotter file, 5-11, 5-20
plotter_is program (Utilities disc), 2-23, 4-2, 4-61 thru 4-62, 5-10, 5-20
Plotters, 1-1, 5-19
address, 5-21
current device, 4-61 thru 4-62, 5-11, 5-20
Plotting in window, 2-21 thru 2-24, 4-45 thru 4-46
Pointer. *See* Display pointer
Port, 5-19
Powering up the system, 4-77 thru 4-79
Print buffer, filling, 4-63 thru 4-64
print command (PAM), 3-24, 4-63 thru 4-64, 5-3, 5-18, 5-19
Print direction, setting, D-16
Print pitch, A-14, D-9 thru D-12
Printer, internal
checking operation, D-20
controlling, D-1 thru D-22
copying screen, 2-10
default modes, D-2
self test, D-20
setting modes, D-1 thru D-22
Printer control program, 4-47 thru 4-48
Printer queue, 3-24
printer_is program (Utilities disc), 4-65 thru 4-66, 5-3, 5-10,5-17 thru 5-19
Printers, 5-3, 5-11, 5-17
setting current device, 4-65 thru 4-66
stopping with *stopprint*, 4-86
Printing files, with *print* command, 4-63 thru 4-64
Printing graphics display, 2-21
Printing screen image, 2-10
print_plot program (Utilities disc), 2-21
Process groups, defined, A-15
Program files, 1-9, 1-11, A-15. *See also* Files
Programmatic access, defined, A-15
Programs
allocated in RAM, 1-12
associated with windows, 1-6
define function keys, 2-8
error for insufficient memory, 1-13
identification of, 4-84 thru 4-85
starting, 3-9
status of, 4-84 thru 4-85
stopping, 1-13, 2-10, 2-18 thru 2-19, 3-10, 4-87
elapsed time of, 4-84 thru 4-85
Purging files, *delete* command, 4-27 thru 4-28

R

RAM
  allocated to programs and files, 1-12
  defined, A-16
  error message when insufficient, 1-13
  increasing available size, 1-13
RAM disc, A-16. *See also* Electronic disc
Random access memory. *See* RAM
Raster scan, printing graphics, D-17
Read permission. *See* Permissions
README files, on discs, 4-6
Redirecting input and output, 3-27 thru 3-30, A-16
Relative path names, 1-11, 3-9, A-16
Removing discs, 5-13
Removing files and folders, *delete* command, 4-27 thru 4-28
Removing HP-IB peripherals, 5-7
rename command (PAM), 4-67 thru 4-68
rename_disc program (Utilities disc), 4-69 thru 4-72
Renaming discs, 4-69 thru 4-72
Renaming files, 3-17, 4-67 thru 4-68
Renaming folders, 4-67 thru 4-68
reread command (PAM), 4-73
Rereading folders, 3-10, 4-73
Resetting computer, 2-10, 2-19, A-16
Resetting internal printer, D-20
ROM, defined, A-16
/rom folder, 3-13
/rom/.environ file, 3-8
Roman Extension character set, 2-6, 2-7, B-4 thru B-5, C-5 thru C-7
Roman8 character set, A-3, A-8, C-1 thru C-7
Root. *See* Root folder
Root folder
  as open folder, 3-11
  contains system folders, 1-11, 3-13
  displayed at power up, 5-3
  contains folders of mounted discs, 5-14
  creating folder in, 1-12
  defined, A-16
  in path name, 1-11
  lost at power down, 1-12
  placing files in, 1-12, 5-17
  related to path names, A-1
  stored in electronic disc, 1-12
  top of hierarchy, 1-10
rpn program (Standard Applications disc), 4-74 thru 4-75
RS-232 device name, 5-18

S

Scaling points, defined, A-17
scan_discs program (in **/rom**), 4-77 thru 4-79, 5-13 thru 5-14
Script files, 3-37 thru 3-39, 5-10, A-17
Scrolling active window, 2-9, 2-14, 2-23
Search efficiency, 3-45
Search paths, 3-44 thru 3-45, 4-42 thru 4-43, A-17
Security for files, 1-11
Self test, internal printer, D-20
send command (PAM), 3-25, 4-80 thru 4-81
Sending text, 3-25, 4-80 thru 4-81
Sequential processing, 3-33 thru 3-35, A-17
Serial interface, 5-2, 5-3, 5-8, 5-9, 5-10, 5-18
  driver for, 4-47
  installation, 4-47

Serial peripherals, 5-8
Serial plotter, 4-61 thru 4-62
set_font program (Standard Applications disc), 4-82 thru 4-83
Setting fonts, *set_font* program, 4-82 thru 4-83
Setting the time, 4-88 thru 4-89
Shells, defined, A-17
Simple file names, defined, A-18
Source, defined, A-18
Sourcing the PAM environment, 4-42 thru 4-44
Spreadsheet program, 1-9
Sprite, defined, A-7
Standard Applications disc, 1-2, 4-1
Starting programs, 3-9
status program (Utilities disc), 4-84 thru 4-85
Stopping programs, 3-10, 4-87
Stopping the printer, 4-86
stopprint command (PAM), 3-25, 4-86
Storing text, 3-10
Strings, defined, A-18
Subfolders, defined, A-18
Subscripts, printing, D-8
Superscripts, printing, D-8
System disc, 1-2, 5-3, 5-10
System keys, purpose, 2-10 thru 2-11
System menu, 2-1, 2-3, 2-12, A-11. *See also* Menus

T

Tab stops, 2-6, 2-20, A-18
Target, defined, A-18
TERM variable, (PAM), 4-42
TERMCAP variable (PAM), 4-42
Terminal devices, defined, A-19
Terminal operation, uses alpha window, A-1
Terminal type (PAM), 4-43
terminate program (Utilities disc), 1-13, 4-87
Text
  changing in *vi* editor, F-8 thru F-14
  defined, A-19
Text editors
  create files, 3-10
  defined, A-19
  using control sequences, D-1
  vi editor, F-1 thru F-16
Text length, setting for printer, D-14 thru D-15
Time, checking and setting, 4-88 thru 4-89
time_of_day program (Utilities disc), 2-13, 4-88 thru 4-89
Time zone, setting, 4-42 thru 4-43
Troubleshooting, *diagnostic* program, 4-29 thru 4-30
Tutor disc, 1-1, 1-2, 1-3
Typing keys, 2-5 thru 2-7
TZ variable (PAM), 4-42

U

U.S. ASCII character set, listing, C-2 thru C-4
Underlined printing, D-11
Unit number, 5-15, A-19
UNIX operating system, 1-1
unmount_disc program (Utilities disc), 4-90 thru 4-91, 5-13, 5-14
Uppercase letters
  in escape sequences, D-3
  in file names, 1-8, 4-3
  typing, 2-5, 2-6, 2-7
User menu, 2-1, A-11. *See also* Menus
User's Guide, 1-1, 1-3
Utilities, 5-9, 5-11, A-19
Utilities disc, 1-2, 4-1

V

Variable assignment, defined, A-20
verify_disc program (Utilities disc), 4-92 thru 4-95
Verifying discs, 4-92 thru 4-95
vi editor, D-1, F-1 thru F-16
view command (PAM), 3-13, 4-96 thru 4-97
Viewing files, 3-23, 4-96 thru 4-97
Volume names, 5-14, A-20
Volume numbers, 5-15, A-20

W

Waiting list, defined, A-15
Window manager programs, defined, A-21
Windows. *See also* Alpha windows; Graphics windows
  active. *See* Active window
  associated with programs, 1-6
  banner defined, A-21
  controlling, 2-11 thru 2-19
  defined, A-21
  displaying, 2-10
  graphics, 5-21
  hiding, 2-16 thru 2-17
  removing, 2-18 thru 2-19
  retrieving hidden, 2-17
  reversing color, 2-15
  saving, 2-16
  sending text to, 4-77 thru 4-78
  shuffling, 2-10, 2-13
  using, 1-5 thru 1-6
Word processor program, 1-2, 1-9
Wrap-around printing, D-16
Write permission. *See* Permissions
Write protection, defined, A-21

Reorder Number
00095-90025

Mfg. No. 00095-90024
Printed in U.S.A.